AD-A032 752   CITY UNIV OF NEW YORK GRADUATE SCHOOL AND UNIV CENTER   F/G 13/13
DEEP BUCKLING OF A THIN OBLATE SPHEROIDAL SHELL UNDER UNIFORM N--ETC(U)
1976   N H JACOBS   AF-AFOSR-2063-71

UNCLASSIFIED   AFOSR-TR-76-1209   NL

1 OF 2
ADA032752

11

752 403

AFOSR - TR - 76 - 1209

12

DEEP BUCKLING OF A THIN OBLATE SPHEROIDAL SHELL UNDER UNIFORM NORMAL PRESSURE

by

NEAL JACOBS

A dissertation submitted to the Graduate Faculty in Mathematics in partial fulfillment of the requirements for the degree of Doctor of Philosophy, The City University of New York.

1976

This manuscript has been read and accepted for the Graduate Faculty in Mathematics in satisfaction of the dissertation requirement for the degree of Doctor of Philosophy.

Date	Chairman of Examining Committee
Date	Executive Officer

Supervisory Committee

The City University of New York

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE

READ INSTRUCTIONS BEFORE COMPLETING FORM

1. REPORT NUMBER: 18 AFOSR - 19 TR - 76 - 1209 2. GOVT ACCESSION NO.	3. RECIPIENT'S CATALOG NUMBER
4. TITLE (and Subtitle): 6 DEEP BUCKLING OF A THIN OBLATE SPHEROIDAL SHELL UNDER UNIFORM NORMAL PRESSURE.	5. TYPE OF REPORT & PERIOD COVERED: Interim
	6. PERFORMING ORG. REPORT NUMBER
7. AUTHOR(s): 10 Neal H. Jacobs 9 Doctoral thesis,	8. CONTRACT OR GRANT NUMBER(s): 15 AF-AFOSR 3-2063-71
9. PERFORMING ORGANIZATION NAME AND ADDRESS: City University of New York, Graduate Center, 33 West 42 St, New York, NY 10036	10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS: 61102F 16 2304 17 A4
11. CONTROLLING OFFICE NAME AND ADDRESS: Air Force Office of Scientific Research/NM, Bolling AFB, Washington, DC 20332	12. REPORT DATE: 11 1976
	13. NUMBER OF PAGES: 94
14. MONITORING AGENCY NAME & ADDRESS (if different from Controlling Office): 12 101p.	15. SECURITY CLASS. (of this report): UNCLASSIFIED
	15a. DECLASSIFICATION/DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT (of this Report)

Approved for public release; distribution unlimited.

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

18. SUPPLEMENTARY NOTES

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

nonlinear elasticity
elastic shells
ellipsoidal shells
blasic stability
buckling of shells
Galerkin method

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

In this paper Eric Reissner's equations (8) for axisymmetrical deformations of thin shells of revolution are specialized to the case of deformation of a thin ellipsoidal shell under uniform normal pressure. The linked pair of non-linear differential equations which results is solved approximately by the Bubnov-Galerkin method, producing the first complete description, including collapse and deep buckling, of an ellipsoidal shell under pressure.

Acknowledgements

This research was conducted under the guidance and with the inspiration of Professor Harry E. Rauch, to whom I owe a deep debt. Partial support for the research came through his grant: Air Force Office of Scientific Research, Office of Aerospace Research, USAF, Grant No. AFOSR 71-2063.

Very substantial support for the research was provided by The City University of New York, in the form of generous allocations for the use of the University Computer Center.

Table of Contents

List of Tables

List of Figures

Section I: Introduction

In this paper Eric Reissner's equations [8] for axisymmetrical deformations of thin shells of revolution are specialized to the case of deformation of a thin ellipsoidal shell under uniform normal pressure. The linked pair of non-linear differential equations which results is solved approximately by the Bubnov-Galerkin method, producing the first complete description, including collapse and deep buckling, of an ellipsoidal shell under pressure.

An oblate ellipsoidal shell is studied because deformation of such a shell is likely to occur axisymmetrically. Oblate shells under pressure have aroused the attention of Clark and Reissner [1], who determined the range of usefulness for such shells of a linear approximation to Reissner's equations, and of Danielson [2], who used special buckling equations to determine buckling pressures for a wide range of geometries. The present paper extends these results through a unified and refinable discussion of all states of a single shell. The comparable numerical results are in reasonable agreement with Danielson's, and a new number is determined -- the lower critical pressure,

the minimum pressure which can sustain the shell in a buckled shape.

Three technical novelties with possible applicability to other problems are worth emphasis. In the Bubnov-Galerkin method as implemented here, sine series rather than appropriate eigenfunction series are substituted for the dependent variables: in effect the vanishing of some integrals is abandoned in favor of obtaining all integrals simply. A powerful addition to the method of continuation, due to Rauch in [6] and useful for taking curves around corners and through loops, is illustrated here as well. And Polak's stable Newton-secant algorithm for solving systems of equations [4] is implemented here and in Rauch et al [6] : tested FORTRAN programs used in both works are included in the appendix.

Section II: Establishment of Equations

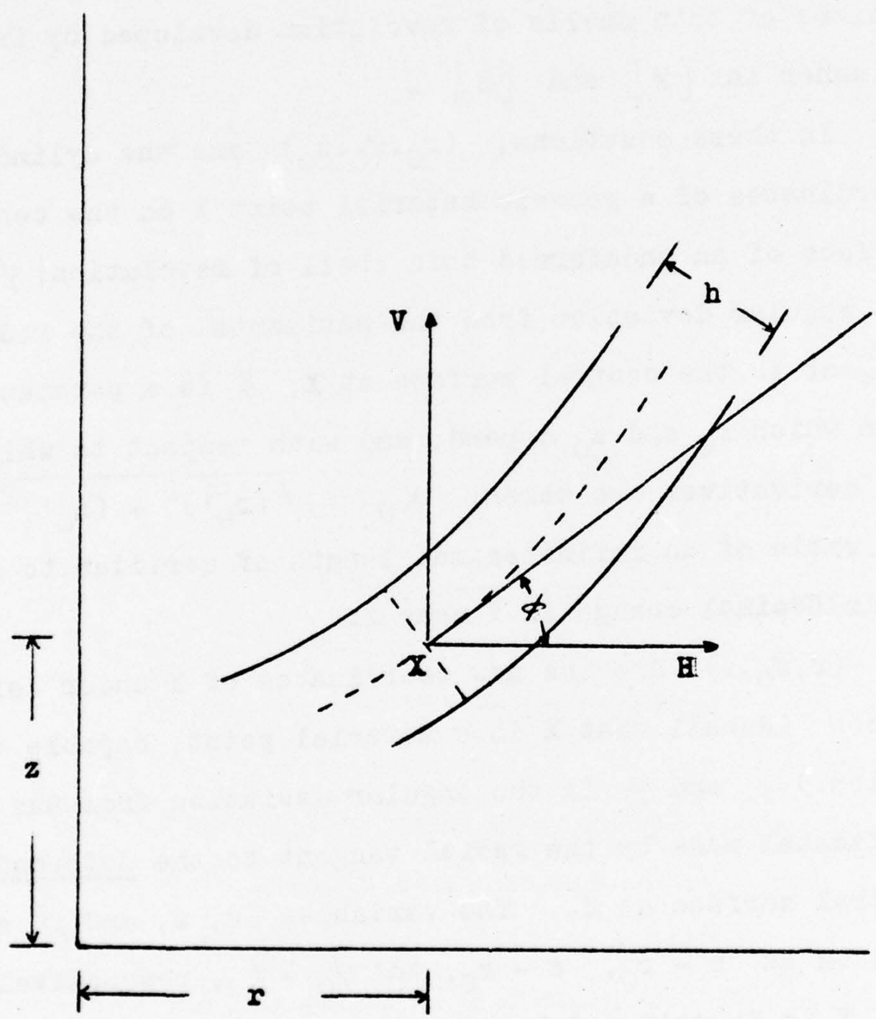
We begin with the equations for axisymmetric deformations of thin shells of revolution developed by Eric Reissner in [7] and [8] .

In these equations, (r_0, θ_0, z_0) are the cylindrical coordinates of a generic material point X on the central surface of an undeformed thin shell of revolution; ϕ_0 is the angular deviation from the horizontal of the radial tangent to the central surface at X; ξ is a parameter upon which r_0 and z_0 depend, and with respect to which all derivatives are taken; $\alpha_0 \equiv \sqrt{(r_0')^2 + (z_0')^2}$ is the ratio of an infinitesimal length of meridian to an infinitesimal change in ξ near X.

(r, θ_0, z) are the new coordinates of X under deformation (Recall that X is a material point, capable of motion.) , and ϕ is the angular deviation from the horizontal made by the radial tangent to the deformed central surface at X. The variables u, w, and β are defined as $r - r_0$, $z - z_0$, and $\phi_0 - \phi$, respectively.

E is Young's modulus for the material of the shell, the ratio of uniaxial stress to the extension it produces; ν is Poisson's ratio for this material, the ratio of transverse contraction to the extension produced by uni-

Figure 1. Cross-section of a piece of deformed shell in the plane $\vartheta = \vartheta_0$



axial stress; h is the thickness of the shell; and C and D are defined to be Eh and $Eh^3/(12\,[1-\nu^2])$.

p_V and p_H are vertical and horizontal components of the pressure on the shell.

Finally, consider surface tractions across infinitesimal surfaces normal to the radial tangent to the central surface; let these be integrated with respect to length along a material line through X and perpendicular to the central surface; let the resultant vector be resolved into a vertical component V and a horizontal component H (Rotational symmetry of shell and deformation are assumed, and so no other component is present.); then Ψ is defined to be $r_0 H$.

With these definitions in mind, we discard all but the three most important terms of second order or higher in β and Ψ from the equations (III) and (IV) of [8] . We also discard one linear term involving ν, β and $r_0{}^2 p_H$. We assume E, h, and ν, and hence C and D, to be constant for the shell, and we multiply through by r_0/α_0. This leaves

$$(1) \quad \frac{r_0}{\alpha_0}\beta'' + \left(\frac{r_0}{\alpha_0}\right)'\beta' -$$

$$\left[\frac{r_0}{\alpha_0}\left(\frac{r_0'}{r_0}\right)^2 - \nu\left(\frac{r_0'}{0}\right)'\right]\beta$$

$$= -\frac{\alpha_0}{D}\Big[\Psi \sin\phi_0 - (r_0 V)\cos\phi_0$$

$$-\beta\left\{\Psi\cos\phi_0 + (r_0V)\sin\phi_0\right\}\Big]$$

$$(2) \quad \frac{r_0}{\alpha_0}\Psi'' + \left(\frac{r_0}{\alpha_0}\right)'\Psi' -$$

$$\left[\frac{r_0}{\alpha_0}\left(\frac{r_0'}{r_0}\right)^2 + \nu\left(\frac{r_0'}{0}\right)'\right]\Psi$$

$$= \alpha_0 C\left[\beta\sin\phi_0 - \frac{1}{2}\beta^2\cos\phi_0\right]$$

$$+ \left[\frac{r_0}{\alpha_0}\,\frac{r_0'z_0'}{r_0^2} + \nu\left(\frac{z_0'}{\alpha_0}\right)'\right](r_0V)$$

$$+ \nu\frac{z_0'}{\alpha_0}(r_0V)' - (r_0^2p_H)'$$

$$- \nu\frac{r_0'}{r_0}(r_0^2p_H)$$

Our present concern is with an ellipsoidal shell of revolution whose central surface is given in cylindrical coordinates by $r_0^2 + z_0^2/b^2 = a^2$. When $b = 1$ the ellipsoid is a sphere, and we desire equations (1) and (2) to take the form of Rauch's [5] (5a) and (5b).

For simplicity we shall choose our unit of length

axial stress; h is the thickness of the shell; and C and D are defined to be Eh and $Eh^3/(12\,[1-\nu^2]\,)$.

p_V and p_H are vertical and horizontal components of the pressure on the shell.

Finally, consider surface tractions across infinitesimal surfaces normal to the radial tangent to the central surface; let these be integrated with respect to length along a material line through X and perpendicular to the central surface; let the resultant vector be resolved into a vertical component V and a horizontal component H (Rotational symmetry of shell and deformation are assumed, and so no other component is present.); then Ψ is defined to be r_0H.

With these definitions in mind, we discard all but the three most important terms of second order or higher in β and Ψ from the equations (III) and (IV) of [8] . We also discard one linear term involving ν, β and $r_0^{\,2}p_H$. We assume E, h, and ν, and hence C and D, to be constant for the shell, and we multiply through by r_0/α_0. This leaves

$$(1)\quad \frac{r_0}{\alpha_0}\,\beta'' + \left(\frac{r_0}{\alpha_0}\right)'\beta' -$$

$$\left[\frac{r_0}{\alpha_0}\left(\frac{r_0'}{r_0}\right)^2 - \nu\left(\frac{r_0'}{0}\right)'\right]\beta$$

so that $a = 1$ and then choose our unit of force so that $E = 1$. Then, with the parameter ξ defined to be ϕ_0 (This parametrization differs from that used in [1] .), the variables appearing in (1) and (2) are given by

$$(3) \quad C \equiv Eh = h$$

$$(4) \quad D \equiv \frac{Eh^3}{12(1-\nu^2)} = \frac{h^3}{12(1-\nu^2)}$$

$$(5) \quad \tan\xi = \frac{dz_0}{dr_0} = -\frac{b^2 r_0}{z_0}$$

$$(6) \quad r_0 = \frac{\sin\xi}{\sqrt{b^2\cos^2\xi + \sin^2\xi}}$$

$$(7) \quad z_0 = \frac{-b^2\cos\xi}{\sqrt{b^2\cos^2\xi + \sin^2\xi}}$$

$$(8) \quad r_0' = \frac{b^2\cos\xi}{(b^2\cos^2\xi + \sin^2\xi)^{3/2}}$$

$$(9) \quad z_0' = \frac{b^2\sin\xi}{(b^2\cos^2\xi + \sin^2\xi)^{3/2}}$$

$$(10) \quad \alpha_0 = \sqrt{(r_0')^2 + (z_0')^2} = \frac{b^2}{(b^2\cos^2\xi + \sin^2\xi)^{3/2}}$$

$$(11) \quad \frac{r_0}{\alpha_0} = \frac{1}{b^2}\sin\xi\,(b^2\cos^2\xi + \sin^2\xi)$$

$$= \sin\xi + \frac{1-b^2}{b^2}\sin^3\xi$$

$$(12) \quad \left(\frac{r_0}{\alpha_0}\right)' = \cos\xi + 3\,\frac{1-b^2}{b^2}\,\sin^2\xi\,\cos\xi$$

$$(13) \quad \frac{r_0'}{\alpha_0} = \cos\xi$$

$$(14) \quad \left(\frac{r_0'}{\alpha_0}\right)' = -\sin\xi$$

$$(15) \quad \frac{z_0'}{\alpha_0} = \sin\xi$$

$$(16) \quad \left(\frac{z_0'}{\alpha_0}\right)' = \cos\xi$$

$$(17) \quad \frac{r_0'}{r_0} = \frac{b^2\cot\xi}{b^2\cos^2\xi + \sin^2\xi}$$

$$(18) \quad \frac{z_0'}{r_0} = \frac{b^2}{b^2\cos^2\xi + \sin^2\xi}$$

$$(19) \quad \frac{r_0}{\alpha_0}\left(\frac{r_0'}{r_0}\right)^2 = \frac{b^2\cos^2\xi}{\sin\xi\,(b^2\cos^2\xi + \sin^2\xi)}$$

$$(20) \quad \frac{r_0}{\alpha_0}\,\frac{r_0'z_0'}{r_0^2} = \frac{b^2\cos\xi}{b^2\cos^2\xi + \sin^2\xi}$$

Under uniform normal pressure ρ, $p_H = -\rho\sin\phi$ and $p_V = \rho\cos\phi$. We use the approximations $p_H = -\rho\sin\xi$ and $p_V = \rho\cos\xi$ and obtain

$$(21) \quad r_0^2 p_H = \frac{-\rho\sin^3\xi}{b^2\cos^2\xi + \sin^2\xi}$$

(22) $(r_0{}^2 p_H)' =$

$$\frac{-\rho(3b^2\sin^2\xi\cos^2\xi + (1 + 2b^2)\sin^4\xi\cos\xi)}{(b^2\cos^2\xi + \sin^2\xi)^2}$$

(23) $(r_0 V)' = -r_0\alpha_0 p_V = \dfrac{-\rho b^2\sin\xi\cos\xi}{(b^2\cos^2\xi + \sin^2\xi)^2}$

(24) $(r_0 V) =$

$$\rho\left\{\frac{b^2}{2(1 - b^2)(b^2\cos^2\xi + \sin^2\xi)} - \frac{1}{2(1 - b^2)}\right\}$$

$$= \frac{-\rho\sin^2\xi}{2(b^2\cos^2\xi + \sin^2\xi)}$$

The peculiar integration constant $-1/(2(1 - b^2))$ was introduced so that all the variables in (3) - (24) approach the corresponding variables in Rauch [5] as b approaches 1.

Substitution of (3) - (24) into (1) and (2) produces

(25) $$\left[\sin\xi + \frac{1 - b^2}{b^2}\sin^3\xi\right]\beta''$$

$$+ \left[\cos\xi + 3\,\frac{1 - b^2}{b^2}\sin^2\xi\cos\xi\right]\beta'$$

$$- \left[\frac{b^2\cos^2\xi}{\sin\xi(b^2\cos^2\xi + \sin^2\xi)} + \nu\sin\xi\right]\beta$$

$$= \frac{12(1 - \nu^2)}{h^3}\left[- \frac{b^2\sin\xi\ \Psi}{(b^2\cos^2\xi + \sin^2\xi)^{3/2}}\right.$$

$$- \frac{\rho\, b^2 \sin^2\xi \cos\xi}{2(b^2\cos^2\xi + \sin^2\xi)^{5/2}} + \frac{b^2\cos\xi \;\beta\Psi}{(b^2\cos^2\xi + \sin^2\xi)^{3/2}}$$

$$\left. - \frac{\rho\, b^2 \sin^3\xi \;\beta}{2(b^2\cos^2\xi + \sin^2\xi)^{5/2}} \right]$$

$$(26) \qquad \left[\sin\xi + \frac{1-b^2}{b^2} \sin^3\xi \right] \Psi''$$

$$+ \left[\cos\xi + 3\,\frac{1-b^2}{b^2} \sin^2\xi \cos\xi \right] \Psi'$$

$$- \left[\frac{b^2\cos^2\xi}{\sin\xi\,(b^2\cos^2\xi + \sin^2\xi)} - \nu \sin\xi \right] \Psi$$

$$= h \left[\frac{b^2 \sin\xi \;\beta}{(b^2\cos^2\xi + \sin^2\xi)^{3/2}} \right.$$

$$\left. - \frac{b^2\cos\xi \;\beta^2}{2(b^2\cos^2\xi + \sin^2\xi)^{3/2}} \right]$$

$$+ \frac{\rho}{(b^2\cos^2\xi + \sin^2\xi)^2} \left[-\tfrac{1}{2} b^2 \sin^2\xi \cos\xi \right.$$

$$+ b^2(3 - \tfrac{1}{2}\nu)\sin^2\xi \cos^3\xi$$

$$\left. + (1 - \tfrac{1}{2}\nu + 2b^2)\sin^4\xi \cos\xi \right]$$

If we let $b = 1$ in equations (25) and (26), divide through by $\sin\xi$, and make the substitution $\varphi = \frac{1}{2}\rho\cos\xi\sin\xi + \Psi$, we obtain the equations (5a) and (5b) of [5] as desired.

Reissner's formulas (10) and (12) of [8] allow the determination of the horizontal and vertical displacements u and w, once β, ϕ, Ψ, Ψ', and (r_0V) are known. In the present case the formulas are

$$(27)\quad u = \frac{1}{h}\left(\frac{r_0}{\alpha_0}\Psi' - r_0^2\rho\sin\phi - \nu\Psi\cos\phi - \nu(r_0V)\sin\phi\right)$$

$$(28)\quad w = \int_0^{\xi}\left[\frac{-\alpha_0\sin\phi}{r_0h}\left\{r_0h + \Psi\cos\phi + (r_0V)\sin\phi - \frac{r_0}{\alpha_0}\nu\Psi' - r_0^2\nu\rho\sin\phi\right\} - \alpha_0\sin\xi\right]d\xi$$

The integral for w is normalized so that the south pole shows no vertical displacement; this integral can be approximated adequately by the trapezoidal rule.

Section III: Algebraization of the Equations

We proceed to find approximations to β and Ψ through the Bubnov-Galerkin method: we replace all the dependent variables and their derivatives by truncated series, $\beta = \sum_{j=1}^{m} B_j \sin j\xi$,

$$\beta' = \sum_{j=1}^{m} j\, B_j \cos j\xi ,$$

$$\beta'' = \sum_{j=1}^{m} (-j^2)\, B_j \sin j\xi ,$$

$$\Psi = \sum_{j=1}^{m} P_j \sin j\xi ,$$

$$\Psi' = \sum_{j=1}^{m} j\, P_j \cos j\xi ,$$

$$\Psi'' = \sum_{j=1}^{m} (-j^2)\, P_j \sin j\xi .$$

This leaves us two equations involving only the parameters h, ρ, b and ϑ, the independent variable ξ, and the 2m unknown constants B_j and P_j. We eliminate ξ by multiplying each equation by $\sin i\xi$ and integrating both sides from 0 to π. If this is done for i from 1 to m, we are left with 2m algebraic equations involving only the 2m unknowns B_j and P_j, the

parameters, and a number of evaluated integrals. It is this system of 2m equations, rather than (25) and (26), which we attempt to solve.

We write this system as equations (44_i) and (45_i) in terms of the integrals S_k and their combinations C_k which we define below.

$$(29)\quad S_1(i,j) = \int_0^{\pi} \sin\xi \sin i\xi \sin j\xi \; d\xi$$

$$(30)\quad S_2(i,j) = \int_0^{\pi} \sin^3\xi \sin i\xi \sin j\xi \; d\xi$$

$$(31)\quad S_3(i,j) = \int_0^{\pi} \cos\xi \sin i\xi \cos j\xi \; d\xi$$

$$(32)\quad S_4(i,j) = \int_0^{\pi} \sin^2\xi \cos\xi \sin i\xi \cos j\xi \; d\xi$$

$$(33)\quad S_5(i,j) = \int_0^{\pi} \frac{\cos^2\xi}{\sin\xi\,(b^2\cos^2\xi + \sin^2\xi)} \sin i\xi \sin j\xi \; d\xi$$

$$(34)\quad S_6(i,j) = \int_0^{\pi} \frac{\sin\xi}{(b^2\cos^2\xi + \sin^2\xi)^{3/2}} \sin i\xi \sin j\xi \; d\xi$$

$$(35)\quad S_7(i) = \int_0^{\pi} \frac{\sin^2\xi \cos\xi}{(b^2\cos^2\xi + \sin^2\xi)^{5/2}} \sin i\xi \, d\xi$$

$$(36)\quad S_8(i,j,k) = \int_0^{\pi} \frac{\cos\xi}{(b^2\cos^2\xi + \sin^2\xi)^{3/2}} \sin i\xi \sin j\xi \sin k\xi \, d\xi$$

$$(37)\quad S_9(i,j) = \int_0^{\pi} \frac{\sin^3\xi}{(b^2\cos^2\xi + \sin^2\xi)^{5/2}} \sin i\xi \sin j\xi \, d\xi$$

$$(38)\quad S_{10}(i) = \int_0^{\pi} \frac{\sin^2\xi \cos\xi}{(b^2\cos^2\xi + \sin^2\xi)^2} \sin i\xi \, d\xi$$

$$(39)\quad S_{11}(i) = \int_0^{\pi} \frac{\sin^2\xi \cos^3\xi}{(b^2\cos^2\xi + \sin^2\xi)^2} \sin i\xi \, d\xi$$

$$(40)\quad S_{12}(i) = \int_0^{\pi} \frac{\sin^4\xi \cos\xi}{(b^2\cos^2\xi + \sin^2\xi)^2} \sin i\xi \, d\xi$$

$$(41)\quad C_1(i,j) = -j^2\left[S_1(i,j) + \frac{1-b^2}{b^2} S_2(i,j)\right] + j\left[S_3(i,j) + 3\,\frac{1-b^2}{b^2} S_4(i,j)\right] - \left[b^2 S_5(i,j) + \nu S_1(i,j)\right]$$

$$(42)\quad C_2(i,j) = C_1(i,j) + 2\nu S_1(i,j)$$

$$(43)\quad C_3(i) = -\frac{1}{2} b^2 S_{10}(i) + b^2 (3 - \frac{1}{2}\nu) S_{11}(i) + (1 - \frac{1}{2}\nu + 2b^2) S_{12}(i)$$

Equations (25) and (26) now give rise to

$$(44_i)\quad \sum_{j=1}^{m} C_1(i,j) B_j = \frac{12(1-\nu^2)b^2}{h^3} \Big[-\frac{1}{2}\rho S_7(i) + \sum_{j=1}^{m} \Big\{ -S_6(i,j) P_j - \Big[\frac{1}{2}\rho S_9(i,j) - \sum_{k=1}^{m} S_8(i,j,k) P_k \Big] B_j \Big\} \Big]$$

$$(45_i)\quad \sum_{j=1}^{m} C_2(i,j) P_j = hb^2 \Big[\sum_{j=1}^{m} \Big\{ S_6(i,j) - \frac{1}{2} \sum_{k=1}^{m} S_8(i,j,k) B_k \Big\} B_j \Big] + \rho C_3(i)$$

Once the S_k and C_k are known for a particular pair of values of b and ν, equations (45_i) express the P_i in terms of the B_i. Thus, solving this system reduces to finding the m variables B_i. We proceed to develop an inexpensive way to compute the S_k and C_k.

Section IV: Evaluation of Integrals

Except for S_5, the integrals S_k are all of the form

$$\int_0^{\pi} (\text{Kernel})(\text{Product of an odd number of sines}) \times (\text{Product of 0 or more cosines})\, d\zeta .$$

Any of these may be evaluated with reasonable economy by a two-step process: (1) use Simpson's Rule to evaluate $\int_0^{\pi} (\text{Kernel}) \sin I\zeta \, d\zeta$ for a sufficiently wide range of values of the integer I; (2) express the S_k in question as a simple function of the integrals so evaluated through use of the formula

(46)

$$\sin I_1 \sin I_2 \ldots \sin I_n \cos J_1 \cos J_2 \ldots \cos J_t$$

$$= \frac{(-1)^{[n/2]}}{2^{n+t-1}} \sum \pm \text{trig}(I_1 \pm I_2 \pm \ldots \pm I_n \pm J_1 \pm J_2 \pm \ldots \pm J_t)$$

In this formula $[n/2]$ is the greatest integer in n/2; "trig" is "sine" if n is odd, "cosine" if n is even; the sum is taken over all 2^{n+t-1} possible combinations of plusses and minusses in the argument of trig; and the sign of trig is the product of the signs of I_1, I_2, . . . I_n in its argument.

For example, to evaluate $S_8(i,j,k)$, first evaluate

$$T(I) = \int_0^{\pi} \sin I\zeta \; d\zeta \, / (b^2 \cos^2\zeta + \sin^2\zeta)^{3/2}$$ for all integers I from $2 - 2m$ to $3m + 1$. Then $S_8(i,j,k) =$

$$-\frac{1}{8}\Big\{ T(i+j+k+1) + T(i+j+k-1) - T(i+j-k+1) - T(i+j-k-1) - T(i-j+k+1) - T(i-j+k-1) + T(i-j-k+1) + T(i-j-k-1)\Big\}.$$

The economy of this process is increased when the symmetries and anti-symmetries of the kernel functions and of $\sin I\zeta$ are considered. All of the kernels and all *odd sines are symmetric* about $\pi/2$; so for I odd, Simpson's Rule need only be applied to half the interval 0 to π. On the other hand, even sines are anti-symmetric about $\pi/2$, so that for I even, $\int_0^{\pi}$ (Kernel) $\sin I\zeta \; d\zeta = 0$, and Simpson's Rule need not be invoked. Finally, *$\sin I\zeta = -\sin(-I)\zeta$* so that Simpson's Rule need only be used on integrals of the form $\int_0^{\pi/2}$ (Kernel) $\sin I\zeta \, d\zeta$ where I is a positive odd integer.

The process described above fails for $S_5(i,j) = \int_0^{\pi} \cos^2\zeta \sin i\zeta \sin j\zeta \, d\zeta \, / (\sin\zeta \, (b^2\cos^2\zeta + \sin^2\zeta))$, since for this integral, formula (46) would suggest an evaluation of integrals of the form $\int_0^{\pi} \cos I\zeta \, d\zeta \, / (\sin\zeta \, (b^2\cos^2\zeta + \sin^2\zeta))$ as a first step. Unfortunately, some of these integrals do not converge, e.g. $b = 1$, $I = 1$. So the integrals S_5 are evaluated by Simpson's Rule directly. Time is

saved however when symmetry considerations lead to the observations: (1) $S_5(i,j) = S_5(j,i)$; and (2) when $i + j$ is odd, $S_5(i,j) = 0$. In computing the integrals S_5, use is made of the formula

$$\frac{\sin I\zeta}{\sin\zeta} = \frac{\sin (I - 1)\zeta}{\sin \zeta} \cos\zeta + \cos (I - 1)\zeta$$

Section V: Polak's Method

In this section, $\bar{x}$ is a vector whose components are $(x_1, x_2, \ldots x_m)$, $\bar{y}$ is a vector whose components are $(y_1, y_2, \ldots y_m)$, etc.

Newton's method for solving a system of m equations in the unknowns $x_1, x_2, \ldots x_m$ requires that each equation be put in the form $f_i(\bar{x}) = 0$. If the guess $\bar{x}$ is near an unknown solution $\bar{y}$, Taylor's Theorem, truncated to exclude non-linear terms, gives

$$f_i\Big|_{\bar{x}} + \frac{f_i}{x_1}\Big|_{\bar{x}} (y_1 - x_1) + \ldots + \frac{f_i}{x_m}\Big|_{\bar{x}} (y_m - x_m) \approx 0$$

Let this linear system be solved for the correction $\bar{z} = \bar{y} - \bar{x}$, and let $\bar{z}$ be added to the original guess $\bar{x}$ resulting in a new vector $\bar{x}^*$. In many cases $\bar{x}^*$ will be significantly closer to the true solution $\bar{y}$ than was $\bar{x}$. If $\bar{x}^*$ is used as a new guess,and if this process is repeatedly applied, and if the original guess was sufficiently near the true solution, convergence to the true solution might possibly occur. When convergence does occur, it is usually rapid.

Polak [4] uses a secant approximation to the first derivatives required by Newton's method, and he keeps Newton's method from violent instability

by requiring that for any new guess the residual $\sqrt{f_1(\bar{x}^*)^2 + f_2(\bar{x}^*)^2 + \ldots + f_m(\bar{x}^*)^2}$ be smaller than the corresponding residual for the last guess, $\bar{x}$ -- else the new guess is not accepted. To keep the method from coming to a halt in such a case, Polak takes advantage of residuals that might as well be computed in the course of computing the first derivatives necessary for Newton's method: for all the f_i's must be computed at

$$(x_1, x_2, \ldots x_k, \ldots x_m) + (0, 0, \ldots \epsilon, \ldots 0) \quad \text{for each } k,$$

and it is possible that such a vector might be an improvement over the guess $\bar{x}$.

Further details and a FORTRAN program for the implementation of Polak's method are in the Appendix.

Section VI: Continuation

Our system of equations involves the parameters b, ρ, h, ν, and m, and the variables B_i. Suppose that for a particular assignment of values to the parameters, a solution for the B_i is known. Then let us take this solution as our initial guess for a new problem in which the assignment of values to one of the parameters, say ρ, is only slightly different from the assignment in the last problem, and otherwise the assignments are the same. If we do so, and apply Polak's method, our chance of finding a solution to the new problem is good. We say we are performing a continuation of the first type, with ρ varying.

There is a new and useful continuation of a second type, in which one of the variables, say B_1, switches roles with one of the parameters, say ρ. Suppose again that for a particular assignment of values to the parameters, a solution for the B_i is known. Then let a new problem be defined by assigning the same values to b, h, ν, and m, varying the former solution for B_1 slightly, and then demanding that a set of values for ρ, B_2, B_3, . . . B_m be found to satisfy the equations.

The second type of continuation has been extremely useful in several situations. The graphs in Figures 2, 3, 4 have long, nearly horizontal stretches which would have been exhorbitantly expensive to obtain by doing a continuation of the first type with ρ varying. Moreover, the loops and near-cusps in these graphs would have been impossible to discover without second-type continuation. Also, transitions from even (solid-line) solutions, in which all the odd-numbered B_i are zero, to odd (dotted-line) solutions, in which some of the odd-numbered B_i are not zero, were effected by continuations of the second type, with an odd-numbered B_i varying. (Odd and even solutions are discussed further in Section VII.) And, in a related problem, when it was found necessary to increase m from a point near the bottom of the curve, second-type continuation with a small unimportant B_i varying hardly at all, had to be combined with first-type continuation with m varying, because of the sensitivity of the system to the necessarily integral jumps in m.

First type continuation was useful in an interesting way in obtaining one of the two starting points used for the development of the curves in Figures 2, 3, 4. In [5] and [6] attention is focussed on deformations

of thin spherical shells, and variables called A_i play roles analogous to the roles of the B_i in the present problem. We shall call the A_i modes in the present discussion. It was expected that in some solutions to equations (A') and (B') of [5], the A_i would gradually increase as i increased, reaching a maximum of A_j for some j, then gradually decrease to a minimum of A_k for some k, then gradually increase again to a relative maximum at A_n for some n, etc., with $|A_j|$ considerably larger than $|A_k|$ which itself would be considerably larger than $|A_n|$, etc. We called A_j the critical mode in the solution, and remarked that as the ratio of radius to thickness $\frac{a}{h}$ increased, j would also increase, and more modes would be necessary to establish an acceptable solution.

It was found, further, that (A') and (B') of [5] could be solved explicitly when the number of modes was 2. However for large $\frac{a}{h}$, attempts to continue, varying the number of modes from 2 up to and past j, met with consistent failure. It seemed that the explicit 2-mode solutions would be useless for producing an acceptable solution to a reasonable problem, involving large $\frac{a}{h}$ -- that is, involving thin shells. Before abandoning the idea, however, we decided to

try starting with an unphysically small $\frac{a}{h} = \frac{1}{2}$, for which the critical mode was A_1, and then alternating first-type continuations, varying $\frac{a}{h}$ for a while, then varying the number of modes, then varying $\frac{a}{h}$ again, never allowing $\frac{a}{h}$ to get so large that it required a critical mode whose subscript was larger than the current number of modes. This worked, and it produced the solution in the first column of Table 1, which is listed in a form translated into the terms of the present paper in the second column. A refinement is listed in the third column.

The solution in the third column of Table 1 was then first-type continued, varying m, then varying b, then varying ρ, then varying b again, to produce a 20-mode starting-point solution for an ellipsoidal shell 1 unit in radius at the equator, 1 unit tall from pole to pole, .02 units thick, and at a pressure of .9E-4 units. This starting-point was continued to produce the dotted-line curves in Figures 2, 3, 4.

The solid-line curves were all continued from the trivial solution $B_i = 0$ for all i to the problems $\rho = 0$, $b = .5$, $\nu = .3$, $m = 20$, 30, or 40.

The solid-line and dotted-line curves all have

common solutions at their highest points. These solutions are listed in Table 2, and may be continued to produce both the solid-line and the dotted-line graphs.

Numbers are listed in the tables in the computer version of scientific notation: in this notation .2910E-3 stands for $.2910 \times 10^{-3}$.

It should be remarked that the close agreement between columns 2 and 3 of Table 1 is a good verification of the consistency of the present paper with [6] and of the correctness of the computer programs used in both projects.

Table 1. Starting-point solution

Solution to (A') and (B') of [5]	Translation to terms of the present paper	Refinement
	$b = 1$	$b = 1$
$\frac{a}{h} = 50$	$h = .02$	$h = .02$
$\rho^* = .6$	$\rho = .2910E-3$	$\rho = .2910E-3$
$A_1 = .2004E-3$	$B_1 = .1888E-2$	$B_1 = .1879E-2$
$A_2 = .3517E-3$	$B_2 = .3693E-2$	$B_2 = .3676E-2$
$A_3 = .5194E-3$	$B_3 = .5731E-2$	$B_3 = .5707E-2$
$A_4 = .7283E-3$	$B_4 = .7496E-2$	$B_4 = .7463E-2$
$A_5 = .9747E-3$	$B_5 = .9716E-2$	$B_5 = .9676E-2$
$A_6 = .1280E-2$	$B_6 = .1138E-1$	$B_6 = .1133E-1$
$A_7 = .1628E-2$	$B_7 = .1362E-1$	$B_7 = .1357E-1$
$A_8 = .2015E-2$	$B_8 = .1480E-1$	$B_8 = .1475E-1$
$A_9 = .2377E-2$	$B_9 = .1651E-1$	$B_9 = .1645E-1$
$A_{10} = .2662E-2$	$B_{10} = .1632E-1$	$B_{10} = .1625E-1$
$A_{11} = .2765E-2$	$B_{11} = .1655E-1$	$B_{11} = .1648E-1$
$A_{12} = .2672E-2$	$B_{12} = .1410E-1$	$B_{12} = .1404E-1$
$A_{13} = .2393E-2$	$B_{13} = .1279E-1$	$B_{13} = .1274E-1$
$A_{14} = .2022E-2$	$B_{14} = .8461E-2$	$B_{14} = .8424E-2$
$A_{15} = .1624E-2$	$B_{15} = .7036E-2$	$B_{15} = .7008E-2$
Residual = .7071E-4	Residual = .1407E-1	Residual = .5513E-4

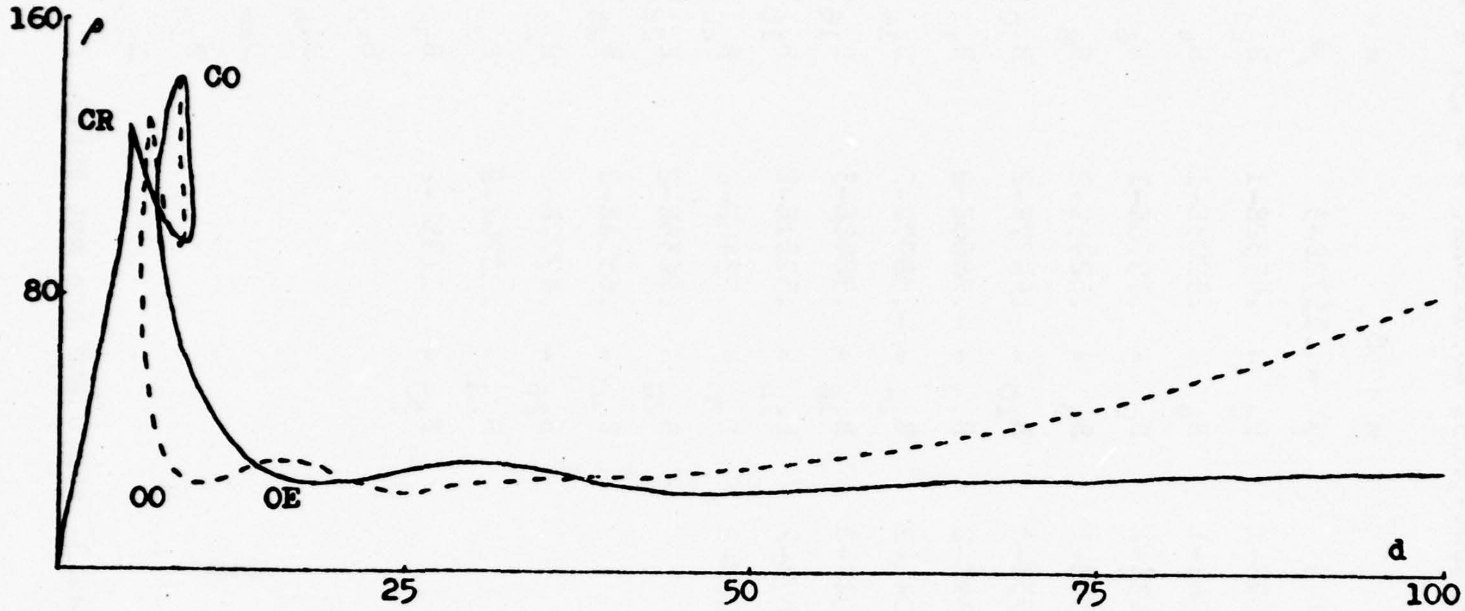
Table 2. Highest-point solutions, b = .5, h = .02,   = .3

	m = 20	m = 30	m = 40
ρ	.1443E-3	.1572E-3	.1524E-3
B_2	.4389E-1	.4552E-1	.4590E-1
B_4	.2076E-1	.3203E-1	.2689E-1
B_6	.1543E-1	.1978E-1	.1599E-1
B_8	.1003E-1	.5293E-2	.5709E-2
B_{10}	.2973E-2	-.4477E-2	-.1614E-2
B_{12}	-.2934E-2	-.6866E-2	-.4335E-2
B_{14}	-.5904E-2	-.3890E-2	-.3279E-2
B_{16}	-.6002E-2	.9848E-3	-.3955E-3
B_{18}	-.4287E-2	.5121E-2	.2587E-2
B_{20}	-.1957E-2	.7345E-2	.4699E-2
B_{22}		.7635E-2	.5678E-2
B_{24}		.6551E-2	.5692E-2
B_{26}		.4777E-2	.5075E-2
B_{28}		.2870E-2	.4154E-2
B_{30}		.1195E-2	.3172E-2
B_{32}			.2275E-2
B_{34}			.1528E-2
B_{36}			.9451E-3
B_{38}			.5105E-3
B_{40}			.1998E-3

Odd-numbered B_i are 0 and are not shown.

Figure 2. Pressure ρ versus deflection d:

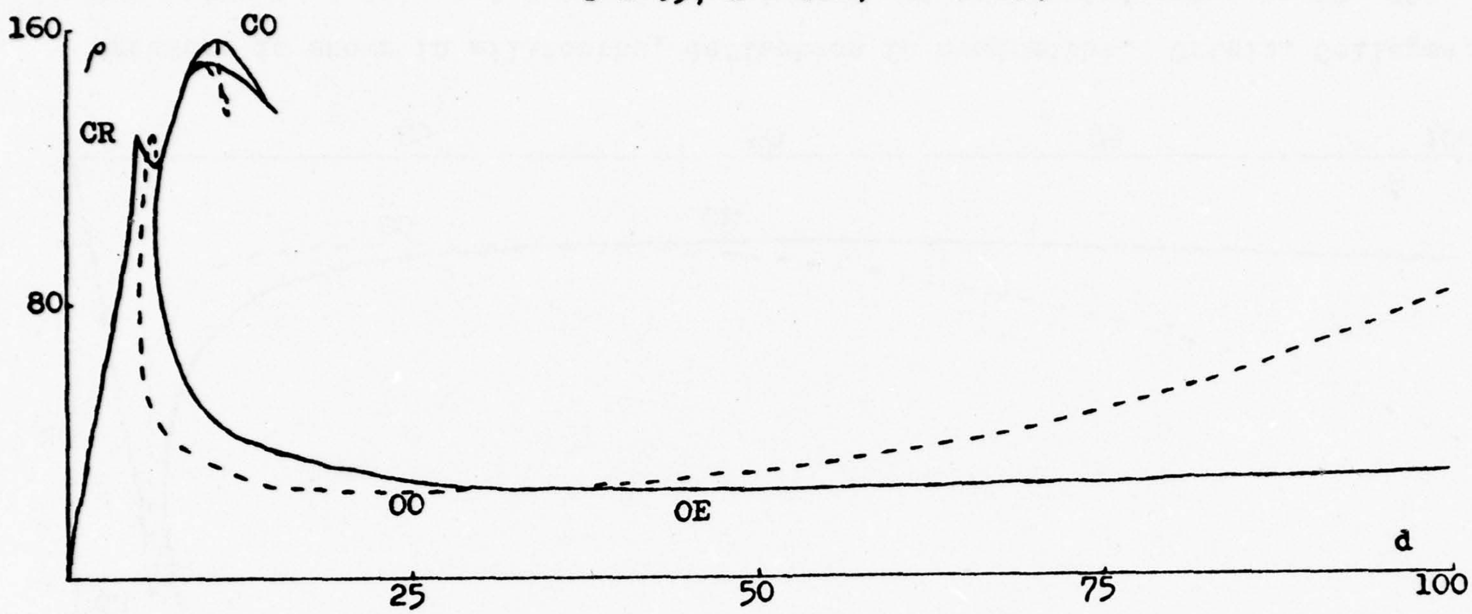
$b = .5,\ h = .02,\ \nu = .3,\ m = 20.$



Pressure is shown in millionths, deflection in hundredths. Crisis, Collapse, Outward-snap (odd), and Outward-snap (even) points are indicated by CR, CO, OO, and OE respectively.

Figure 3. Pressure ρ versus deflection d:

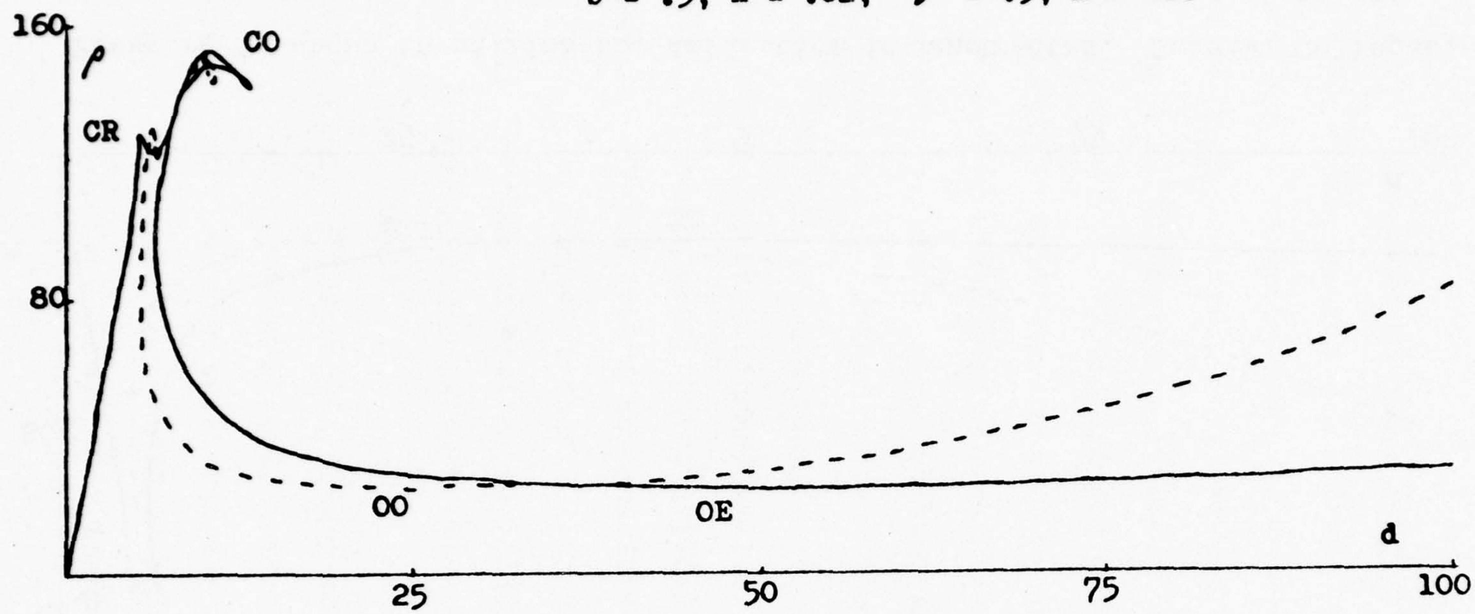
$b = .5$, $h = .02$, $\nu = .3$, $m = 30$.



Pressure is shown in millionths, deflection in hundredths. Crisis, Collapse, Outward-snap (odd), and Outward-snap (even) points are indicated by CR, CO, OO, and OE respectively.

Figure 4. Pressure ρ versus deflection d:

$b = .5$, $h = .02$, $\nu = .3$, $m = 40$.



Pressure is shown in millionths, deflection in hundredths. Crisis, Collapse, Outward-snap (odd), and Outward-snap (even) points are indicated by CR, CO, OO, and OE respectively.

Section VII: Interpretation

An even solution to this system of equations is one in which $\beta = \sum_{i=1}^{m} B_i \sin i\xi$ is anti-symmetric about $\frac{\pi}{2}$. This state is characterized by a symmetric alignment of the shell with respect to the equatorial plane, and by the vanishing of the odd-numbered B_i. A solution which is not even is called odd. In Figures 2, 3, 4 the odd solutions are indicated by dotted lines, the even solutions by solid ones.

For physical reasons, any odd solution must have a matching odd solution in which the deformed shell looks like the original deformed shell turned upside-down. This would be effected by a matching solution in which the even-numbered B_i are the same as in the original solution, but the odd-numbered B_i are minus what they were in the original solution. Any odd solution and its mate produce the same deflection and so are represented by the same point in Figure 2, 3, or 4.

The graph of the odd solutions may be expected to have an even solution as one endpoint. It is of great interest that this even solution happens to be the highest point of the figure: strings of odd solutions have been continued up to such points by

continuations of the second type, and have been found to transform themselves smoothly either into strings of even solutions, or through a single even solution into strings of matching odd solutions -- depending upon whether the particular B_i varying was even- or odd-numbered. Conversely, a string of even solutions for the problem $m = 20$, $b = .6$, $\nu = .3$, $h = .02$ has been transformed at its highest point into a string of odd solutions by varying an odd-numbered B_i.

Deflection, shown on the horizontal axis in Figures 2, 3, 4, is defined in this paper as the difference between the lengths of the polar axes of the undeformed and the deformed shells. Since the poles cannot pass through each other, deflection cannot be greater than the length of the undeformed polar axis, and so our graphs stop at this length. But mathematical solutions with this unphysical characteristic do exist: the graphs seem to continue to rise to the right beyond the realistic cut-off.

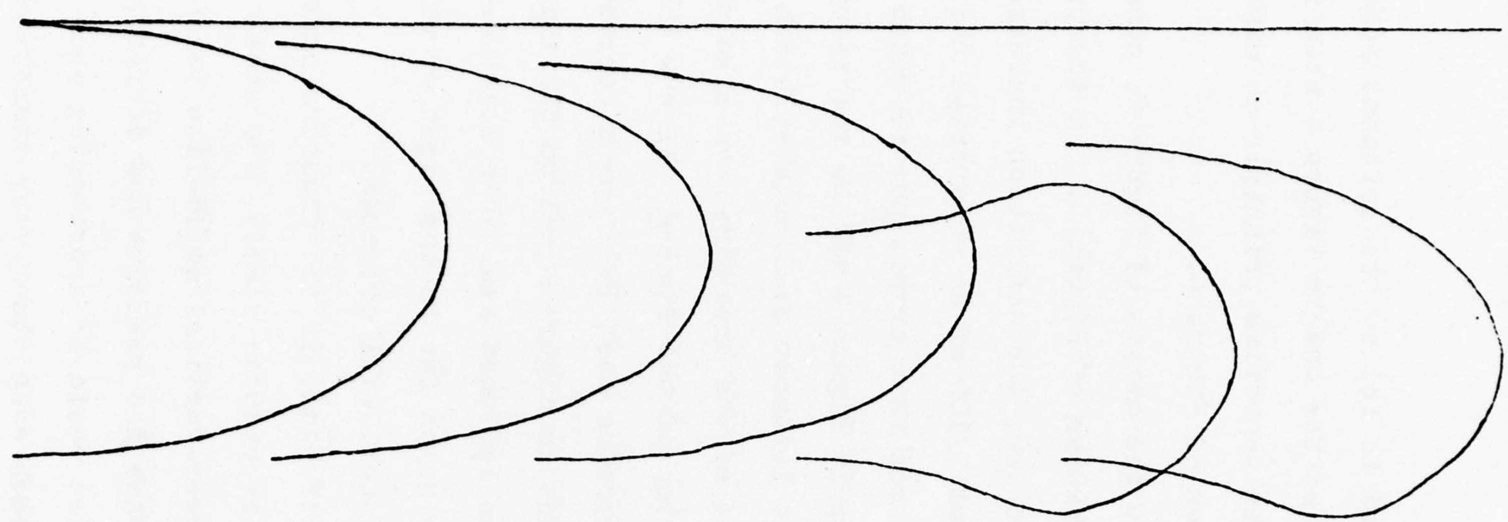
We have not attempted an energy analysis of our solutions. But it seems plausible that under a given pressure, a shell with a substantially larger deflection possesses more elastic potential energy than a shell with a smaller deflection. Under this assumption, the shell

would be expected to follow the leftmost accessible branch of such a curve as the one in Figure 4 when in an unstable state. With this tentative principle in mind we offer an interpretation of Figure 4.

We begin at the origin of Figure 4, corresponding to the first meridian of Figure 5. As the pressure is raised from 0 to 129, the deflection increases almost linearly to about 5/100 of an equatorial radius. If the pressure passes this crisis point, which corresponds to the near-cusp in Figure 4 and to the second meridian of Figure 5, the leftmost available solution will be on the left side of the breaking-wave-like piece of graph near the top of the figure. If now the pressure continues to increase past 152, which corresponds to the highest point in Figure 4 and to the third meridian of Figure 5, the leftmost available solution will be on the dotted-line graph far to the right of what has been drawn, and the shell will collapse.

On the other hand, if the pressure passes the crisis point and then is reduced slowly, the shell might find itself on the near-vertical solid-line below the breaking wave. By the time the pressure had decreased past 80, the shell's poles would be approaching each other visibly, and would approach each other very dramatically as the

Figure 5. Five meridians: $b = .5$, $h = .02$, $\nu = .3$, $m = 40$.



The meridians correspond to the origin and to the points labelled CR, CO, OE, and OO of Figure 4. Deflection is measurable from the line at the top.

pressure dropped from 40 to 25: the deflection would increase from 14/100 to 48/100 in that range. If the pressure then dropped below 25, the only available solution would be on the part of the graph near the origin -- in effect, the shell would snap out from the shape indicated by the fourth meridian in Figure 5 to a shape close to the undeformed shape of the first meridian.

If instability is a feature of the solutions on the section of solid-line graph we have just been discussing, then at any disturbance the shell might leave that solid-line graph, with its symmetric north-south indentations, and seek the lower-potential further-left dotted-line graph with its one-pole indentation. Suppose that so: then as the pressure is reduced from 40 to 25, the deflection would increase from 8/100 to 26/100 before snapping out from the shape shown in the fifth meridian of Figure 5.

It is remarkable that the bottom halves of the fourth and fifth meridians of Figure 5 match so well, and that in Figure 4 the even graph from (14,40) rightwards is a half-speed imitation of the odd graph from (8,40) rightwards. (A similar comment holds for Figures 2 and 3 as well, and for graphs obtained in

Rauch et al [6] for spherical shells.) It is remarkable also that the snap-out pressures on the odd and even graphs in Figure 4 agree to three places: it seems reasonable to take either number, as listed in Table 3, to be our approximation to the lower critical pressure, the least pressure which can maintain a buckled shape of the shell.

Danielson's buckling pressure for the oblate ellipsoidal shell of our work is .137E-3 as indicated in Figure 10 of [2] . This corresponds well to our crisis pressure, .1297E-3.

Our collapse pressure and our lower critical pressure are 118% and 19% respectively of our crisis pressure. The first onset of substantial non-convexity occurs as the pressure decreases (after having passed the crisis pressure) in the range 71% - 67% of the crisis pressure.

The programs which performed the work of this paper were run on an IBM 370/168 at the City University of New York's University Computer Center. The even $m = 40$ graph was obtained using 100K and 50 minutes; the odd $m = 40$ graph required 340K and 170 minutes. If further work shows that the most important pressures -- the crisis pressure, the collapse pressure, and

Table 3. Important Pressures:

$b = .5$, $h = .02$, $\nu = .3$

	m = 20	m = 30	m = 40
Crisis	.1302E-3	.1300E-3	.1297E-3
Collapse	.1443E-3	.1572E-3	.1524E-3
Outward-snap (even)	.2576E-4	.2485E-4	.2488E-4
Outward-snap (odd)	.2470E-4	.2478E-4	.2481E-4

These pressures are given in the units of the paper. To convert to real units, multiply by Young's modulus for the material in question. For example, if a shell fitting the description $b = .5$, $h = .02$ is made of a material for which $\nu = .3$ and $E = 30,000,000$ pounds per square inch, then such a shell would collapse at a pressure near 4572 pounds per square inch, and would be incapable of sustaining a buckled shape at pressures below 744 pounds per square inch.

snap-out pressure -- all continue to be obtainable from the even graphs alone, then future work on the question of deformation of ellipsoidal shells may concentrate on developing only these much more economical graphs. Future work should also attempt to include more of the terms of E. Reissner's differential equations.

Appendix: FORTRAN Programs

This appendix serves primarily as a place of deposit for the FORTRAN programs which developed the data used in the present paper and in Rauch et al [6]. These programs fall into two classes: major programs which utilize a particular subprogram SECANT to solve systems of equations; and satellite programs which compute necessary tables of integrals, or which produce tables of points describing deformed shells, or which translate solutions from [6] to solutions to the same problem as defined in this paper.

The second purpose of this appendix is to describe and illustrate SECANT. This description is especially non-rigorous and should be understood as a bag of tricks rather than as scientific fact.

SECANT was modelled on Polak [4] by Rauch, implemented by Marz, and modified again by the present author in Rauch et al [6]. It is designed to be used with a subprogram GAUSS, modelled on the IBM SSP program SIMQ [3], which solves systems of <u>linear</u> equations, and with a subprogram RESIDU which calculates the goodness of a guessed solution to a system of equations by substituting the guess into the system, and taking the square root

of the sum of the squares of the differences between the left and right sides of each equation in the system. If this square root RESID were ever zero, the guess would be a genuine solution to the system. In practice, it is usually sufficient for RESID to be less than an agreed-upon small number.

The subprograms SECANT, GAUSS, and RESIDU communicate through a block of COMMON data and through argument lists. In the programs below the author has restricted the common data to items of REAL type and the argument lists to items of INTEGER type. Except for variations in the COMMON block and in the argument lists, and in SECANT's DIMENSION statement, SECANT and GAUSS are immutable from application to application, and are listed below only once. RESIDU, on the other hand, must be redesigned for each application to produce the differences between the two sides of each equation (stored in one column of a two-column matrix E) and to produce the residual RESID.

SECANT starts off with a central guessed solution and tries to improve that guess in two ways. First it takes additional guesses by modifying each coordinate of the original guess, one at a time, by a fixed amount EPSILO. If any of these new guesses produces a smaller

residual, that new guess becomes the center for further guessing, and we say we have a local variation improvement. At the same time that it is taking these guesses, however, SECANT is building up, through the matrix E, a table HBAR of approximate first partial derivatives for use in creating a better-aimed guess of a Newton-method type. After an initial period of relying upon local variation alone (while the table of approximate partials is being developed), SECANT alternates a local-variation guess with a Newton guess, and does not accept a local-variation improvement unless the Newton guess of the same iteration fails to produce an improvement.

We note that each time an improvement is adopted, the central guess is changed and the table of approximate partials for the next Newton guess will have most of its columns thrown slightly out of alignment. This potential difficulty, however, is usually no difficulty at all, and indeed, once the Newton guessing starts producing improvements, it usually continues producing improvements rapidly until the central guess is so good that the old EPSILO is too large to produce meaningful partials anymore.

The key to efficient use of SECANT is an efficient

subprogram RESIDU. In applications of any complexity, machine time is spent chiefly in calculation of residuals and data for partials. In the real programs which follow the program illustrating the use of SECANT, every non-obscuring device available to the author to save time in RESIDU has been utilized.

It is also vital that RESIDU not be called unnecessarily. That is the reason why no Newton guesses are tried until the table of approximate partials has been filled with reasonable data, and why the Newton guess is only tried once and not repeatedly cut down and re-tried as Polak suggests. That is also the reason why DTEST, which helps control the size of EPSILO, is kept large, and why ZTEST, which tests the size of the residual, is kept flexible, so that guesses which are good enough shall be accepted as solutions promptly, and the machine shall not waste its time in negligible and expensive improvements.

The real programs which use SECANT have built into their main programs and their RESIDU subprograms the ability to handle continuations of both types described in Section VI above; this feature has proved economical.

A feature that has proved wasteful has been the ability to handle both odd and even computations (as

defined in Section VII above). Storage requirements for the tables necessary for odd computation are extravagant when even computation is planned; and features in RESIDU that save time for odd computation can waste time for even.

We proceed to the example used in the illustrative program. It is proposed to solve the system

$$
\begin{aligned}
x^2 - pxy + 3y^2 - 2xz &= 30 \\
w - 2xz + pwyz^2 &= -20 \\
pwx - 3z + zy^2 - x^5 &= 19 \\
3w^2 - 2xw + 7z^3 &= 96
\end{aligned}
$$

In this system, p is a parameter whose initial value is 2.

This system was created artificially around the solution $w = 4$, $x = 1$, $y = -3$, $z = 2$, but we start off in the main program with an initial guess of 0, 1, 0, .5 . (In the main program, M is the number of equations and variables, ZPERM is a vector that holds the guess, and PARAM is p.) What happens during the run is illustrative of the strengths and weaknesses of the procedure.

SECANT fails to solve this initial problem, stopping

after 172 calls to RESIDU at w = 4.30, x = 1.27, y = -1.13, z = 1.93, which is associated with the large residual 33.24 . SECANT then passes control back to the main program which does a type one continuation on the parameter PARAM. For PARAM = 2.01, and with w, x, y, and z starting at 4.30, 1.27, -1.13, and 1.93, SECANT <u>succeeds</u> in 51 calls to RESIDU in obtaining the solution w = 3.93, x = 1.11, y = -2.98, z = 2.03, which is associated with the small residual .0000890 . The continuation now proceeds to find equally good solutions in equally few calls to RESIDU, for as many new values of PARAM as time permits.

When the solution PARAM = 2.01, w = 3.93, x = 1.11, y = -2.98, z = 2.03 is inserted as the new starting point in a later run, and the continuation is directed backwards, the following solution is obtained for PARAM = 2.00 -- w = 3.96, x = 1.10, y = -2.98, z = 2.02, residual = .0000458 . We are surprised to find this distinct bona fide solution in such close proximity to the known solution w = 4, x = 1, y = -3, z = 2 -- the solution we had hoped to find. It may be that some combination of continuations of first and second type can take us from the solution we have to the one we wished to obtain. But it must be remarked that in a

less artificial situation we might have no idea that another solution was nearby.

ILLUSTRATIVE PROGRAM -- USE OF THE SUBROUTINE SECANT.

```
C       MAIN ROUTINE
C   THIS ROUTINE READS IN THE INITIAL DATA, CALLS SECANT,
C       AND MAKES DECISIONS ABOUT WHETHER AND HOW TO CONTINUE
C       ONCE SECANT SUCCEEDS OR FAILS IN ITS WORK.
      COMMON ZTEMP(10),ZPERM(10),DELTA,RESID,
     C  E(2,10),FF(10,11),DTEST,ZTEST,PARAM
C   IF THE DIMENSION OF ZPERM IS (M),  THEN THE DIMENSIONS
C       OF ZTEMP, E, AND FF ARE (M), (2,M), AND (M,M+1)
C       RESPECTIVELY.
C
C   INPUT DATA HERE.
      M=4
      ZPERM(1)=0
      ZPERM(2)=1
      ZPERM(3)=0
      ZPERM(4)=.5
      PARAM=2
C   COMPUTE OR READ IN SPECIAL CONSTANTS HERE
C   (THIS PROBLEM REQUIRES NO SPECIAL CONSTANTS.)
C   THE MAIN LOOP BEGINS HERE
    1 ZTEST=.1E-3
      DTEST=.6E-04
      DELTA=1
      DO 13 I=1,M
   13 IF (DELTA.LT.ABS(ZPERM(I))) DELTA=ABS(ZPERM(I))
      CALL SECANT(M)
      CALL REPORT(M)
C   A PARAMETER IS VARIED  AND THE PROGRAM IS CYCLED BACK.
      PARAM=PARAM+.01
      IF (PARAM.LT.(2.1)) GO TO 1
      STOP
      END
```

ILLUSTRATIVE PROGRAM -- USE OF THE SUBROUTINE SECANT.

```
      SUBROUTINE SECANT (M)
C  THIS ROUTINE ATTEMPTS TO SOLVE THE SYSTEM OF EQUATIONS.
      COMMON ZTEMP(10),ZPERM(10),DELTA,RESID,
     C E(2,10),FF(10,11),DTEST,ZTEST,PARAM
      REAL NU
      DIMENSION OMEGA(10),ETEMP(10),D(20),DEL(10),HBAR(10,10
     C                                  ),RTEST(10)
C  THE DIMENSIONS OF OMEGA, ETEMP, AND DEL ARE ALWAYS  THE
C       SAME AS THAT OF ZPERM.  THE DIMENSION OF D IS TWICE
C       THAT OF ZPERM, AND THAT OF HBAR IS THAT OF ZPERM
C       SQUARED.  THE DIMENSION OF RTEST IS ALWAYS 10.
C
C  MEANINGS OF VARIABLES.
C       ZPERM IS THE VECTOR CONTAINING THE BEST GUESS SO FAR
C            FOR THE SOLUTION OF THE EQUATIONS.  IT IS
C            ASSOCIATED WITH THE RESIDUAL RPERM.
C       ZTEMP IS THE CURRENT GUESS VECTOR. ITS RESIDUAL IS
C            RTEMP.
C       OMEGA IS A VECTOR WHICH CONTAINS A BETTER GUESS THAN
C            THE CURRENT ZPERM.  THIS VECTOR CAN ONLY BE FOUND
C            BY A LOCAL VARIATION.  OMEGA IS NOT AUTOMATICALLY
C            SUBSTITUTED FOR ZPERM.  SUCH A SUBSTITUTION ONLY
C            OCCURS IF NEWTON'S METHOD DOES NOT ALSO PRODUCE
C            IMPROVEMENT IN THE SAME ITERATION.  THE RESIDUAL
C            ASSOCIATED WITH OMEGA IS RRTEMP.
C
C       E AND ETEMP CONTAIN VALUES OF THE LEFT-HAND-SIDES
C            OF THE M EQUATIONS TO BE SOLVED.  E(1,J) IS
C            ASSOCIATED WITH ZPERM, E(2,J) WITH ZTEMP, AND
C            ETEMP WITH OMEGA.
C      DEL AND HBAR CONTAIN FIRST DIFFERENCES FOR USE IN
C            NEWTON'S METHOD.
C
C
C  FIRST THE PARAMETERS OF THE COMBINED LOCAL-VARIATION
C       -AND-NEWTON-METHOD ARE INITIALIZED.  SOME OF THIS
C       INITIALIZATION IS DONE BY THE MAIN PROGRAM ALREADY.
C       THE PARAMETERS SO PREPARED ARE DELTA, ZTEST
C       AND DTEST.
      KOUNT=0
      WRITE (6,19) DELTA,KOUNT
   19 FORMAT (7X,'DELTA IS NOW ',E10.3,6X,I5)
      DO 21 I=1,10
   21 RTEST(I)=1.0E60
      NU=1000000
      KSUM=0
      J=0
      IFLAG=0
```

ILLUSTRATIVE PROGRAM -- USE OF THE SUBROUTINE SECANT.

```
      DO 20 I=1,M
      D(I)=1
   20 D(M+I)=-1
C  WE BEGIN BY ESTABLISHING DATA FOR THE FIRST GUESS.
  100 DO 110 I=1,M
  110 ZTEMP(I)=ZPERM(I)
      CALL RESIDU(1,M)
      KOUNT=KOUNT+1
      RPERM=RESID
      WRITE (6,1404) RPERM,KOUNT
      IF (RPERM.LT.ZTEST) GO TO 1400
C  THE ITERATION BEGINS HERE.  FIRST COMES AN ATTEMPT TO
C      IMPROVE THE CURRENT GUESS BY VARIATION OF A SINGLE
C      VARIABLE.  IF THIS SUCCEEDS, THE IMPROVEMENT IS
C      STORED TEMPORARILY IN OMEGA AND ITS ASSOCIATES.
C      AT THE SAME TIME, DATA IS BEING GATHERED FOR THE
C      BENEFIT OF NEWTON'S METHOD.
  200 IF (J.EQ.2*M) J=0
      J=J+1
  300 EPSILO=AMIN1(DELTA,NU)
  400 JJ=J- ((J-1)/M)*M
      DO 405 I=1,M
  405 ZTEMP(I)=ZPERM(I)
      ZTEMP(JJ)=ZPERM(JJ)+EPSILO*D(J)
      CALL RESIDU(2,M)
      KOUNT=KOUNT+1
      RTEMP=RESID
      IF (RTEMP.GE.ZTEST) GO TO 500
      ZPERM(JJ)=ZTEMP(JJ)
      RPERM=RTEMP
      GO TO 1400
  500 DO 510 I=1,M
      DEL(I)=(E(2,I)-E(1,I))/EPSILO
  510 HBAR(I,JJ)=DEL(I)*D(J)
  600 IF (RTEMP.GE.RPERM) GO TO 690
      IFLAG=1
  620 DO 630 I=1,M
      ETEMP(I)=E(2,I)
  630 OMEGA(I)=ZTEMP(I)
      RRTEMP=RTEMP
  690 IF (KOUNT.LE.M) GO TO 1200
C  NOW NEWTON'S METHOD IS GIVEN A CHANCE TO TRY TO IMPROVE
C      THE GUESS.   IF IT SUCCEEDS, ITS SUCCESS IS GIVEN
C      PRECEDENCE EVEN OVER A LARGER SUCCESS OF THE METHOD
C      OF LOCAL VARIATION.  IF IT FAILS, THE LOCAL-VARIATION
C      IMPROVEMENT STORED IN OMEGA -- IF THERE WAS INDEED
C      A LOCAL-VARIATION IMPROVEMENT -- IS ADOPTED.  ELSE
C      NO CHANGE TAKES PLACE IN ZPERM.   IN ANY OF THESE
```

ILLUSTRATIVE PROGRAM -- USE OF THE SUBROUTINE SECANT.

```
C       THREE CASES, THE ITERATION BEGINS AGAIN AT LINE 200.
  700 DO 710 I=1,M
      DO 710 II=1,M
  710 FF(I,II)=HBAR(I,II)
      DO 720 I=1,M
  720 FF(I,M+1)=E(1,I)
      CALL GAUSS(M,KS)
      IF (KS.EQ.1) GO TO 1198
  900 DO 910 I=1,M
  910 ZTEMP(I)=ZPERM(I)-FF(I,M+1)
      CALL RESIDU(2,M)
      KOUNT=KOUNT+1
      RTEMP=RESID
      IF (RTEMP.GT.ZTEST) GO TO 1000
      DO 920 I=1,M
  920 ZPERM(I)=ZTEMP(I)
      RPERM=RTEMP
      GO TO 1400
 1000 IF (RTEMP.GT.RPERM) GO TO 1200
      IFLAG=0
      DO 1010 I=1,M
      ZPERM(I)=ZTEMP(I)
 1010 E(1,I)=E(2,I)
      RPERM=RTEMP
      WRITE (6,1403) RPERM,KOUNT
      NU=0
      DO 1020 I=1,M
 1020 NU=NU+FF(I,M+1)**2
      NU=NU**.5
      GO TO 200
 1198 KSUM=KSUM+1
      WRITE (6,1199) KSUM
      IF (KSUM.EQ.M) GO TO 1400
 1199 FORMAT (7X,'GAUSS BAILED OUT',6X,I5)
 1200 IF (J.LT.2*M) GO TO 1300
      DELTA=DELTA/2
      IF (DELTA.LT.DTEST) GO TO 1400
      WRITE (6,19) DELTA,KOUNT
 1300 IF (IFLAG.EQ.0) GO TO 200
      IFLAG=0
      DO 1310 I=1,M
      E(1,I)=ETEMP(I)
 1310 ZPERM(I)=OMEGA(I)
      RPERM=RRTEMP
      WRITE (6,1402) RPERM,KOUNT
      DO 1330 I=1,9
 1330 RTEST(11-I)=RTEST(10-I)
      RTEST(1)=RPERM
```

```
      IF (RTEST(1)/RTEST(10).LE.(.99)) GO TO 200
C   THERE ARE SEVERAL WAYS FOR THIS SUBROUTINE TO TERMINATE.
C      A SUCCESSFUL TERMINATION OCCURS IF RPERM GETS SMALLER
C      THAN ZTEST. UNSUCCESSFUL TERMINATIONS OCCUR WHEN DELTA
C      GETS TOO SMALL, OR WHEN TEN LOCAL VARIATIONS IN A ROW
C      PRODUCE LESS THAN A 1 PER CENT CHANGE IN THE RESIDUAL,
C      OR WHEN GAUSSIAN REDUCTION FAILS TO PRODUCE A TEST
C      VECTOR FOR NEWTON'S METHOD FOR THE M-TH TIME.
 1400 WRITE (6,1404) RPERM,KOUNT
 1402 FORMAT (6X,E14.7,6X,I5,6X,'LOCAL VARIATION')
 1403 FORMAT (6X,E14.7,6X,I5,6X,'NEWTON')
 1404 FORMAT (6X,E14.7,6X,I5)
      RETURN
      END
```

ILLUSTRATIVE PROGRAM -- USE OF THE SUBROUTINE SECANT.

```
      SUBROUTINE GAUSS (M,KS)
C     THIS ROUTINE SOLVES SYSTEMS OF LINEAR EQUATIONS.
      COMMON ZTEMP(10),ZPERM(10),DELTA,RESID,
     C  E(2,10),FF(10,11),DTEST,ZTEST,PARAM
C      GAUSSIAN REDUCTION, WITH PIVOTING.
      MM=M+1
      KS=0
      TOL=.1E-15
C      FORWARD SOLUTION, WORK ON THE J-TH COLUMN.
      DO 65 J=1,M
      JJ=J+1
      BIGA=0
C      FIND THE ROW WHOSE J-TH COEFFICIENT IS LARGEST.
      DO 30 I=J,M
      IF (ABS(BIGA)-ABS(FF(I,J))) 20,30,30
   20 BIGA=FF(I,J)
      IMAX=I
   30 CONTINUE
C      IF THE BIGGEST J-TH COEFFICIENT IS TOO SMALL,
C         THEN GIVE UP.
      IF (ABS(BIGA)-TOL) 35,35,40
   35 KS=1
      RETURN
C      MOVE THE ROW WITH THE BIGGEST J-TH COEFFICIENT TO THE
C         J-TH POSITION, REPLACE IT BY THE FORMER J-TH ROW.
   40 DO 45 K=1,MM
      SAVE=FF(J,K)
      FF(J,K)=FF(IMAX,K)
   45 FF(IMAX,K)=SAVE
C      DIVIDE THE J-TH ROW BY ITS LEADING COEFFICIENT.
      DO 50 K=1,MM
   50 FF(J,K)=FF(J,K)/BIGA
C      ELIMINATE THE J-TH VARIABLE FROM THE REMAINING EQUATIONS
      IF (J.EQ.M) GO TO 65
      DO 60 I=JJ,M
      DO 60 K=JJ,MM
   60 FF(I,K)=FF(I,K)-FF(I,J)*FF(J,K)
   65 CONTINUE
C      BACKWARD SOLUTION, COMPUTE THE N-TH VALUE INTO FF(N,M+1)
      DO 70 J=2,M
      JJ=M+1-J
      JJJ=J-1
      DO 70 I=1,JJJ
      II=MM-I
   70 FF(JJ,MM)=FF(JJ,MM)-FF(JJ,II)*FF(II,MM)
      RETURN
      END
```

ILLUSTRATIVE PROGRAM -- USE OF THE SUBROUTINE SECANT.

```
      SUBROUTINE RESIDU(NAME,M)
C   THIS ROUTINE PREPARES A RESIDUAL AND THE ARRAY E FOR
C        THE SUBROUTINE SECANT.
      COMMON ZTEMP(10),ZPERM(10),DELTA,RESID,
     C  E(2,10),FF(10,11),DTEST,ZTEST,PARAM
      W=ZTEMP(1)
      X=ZTEMP(2)
      Y=ZTEMP(3)
      Z=ZTEMP(4)
      E(NAME,1)=X**2-PARAM*X*Y+3*Y**2-2*X*Z-30
      E(NAME,2)=W-2*X*Z+PARAM*W*Y+Z**2+20
      E(NAME,3)=PARAM*W*X-3*Z+Z*Y**2-X**5-19
      E(NAME,4)=3*W**2-2*X*W+7*Z**3-96
      RESID=E(NAME,1)**2+E(NAME,2)**2+E(NAME,3)**2+E(NAME,4)
                                                  **2
      RESID=RESID**.5
      RETURN
      END
```

ILLUSTRATIVE PROGRAM -- USE OF THE SUBROUTINE SECANT.

```
      SUBROUTINE REPORT(M)
C  THIS ROUTINE WRITES UP THE OUTPUT.
      COMMON ZTEMP(10),ZPERM(10),DELTA,RESID,
     C  E(2,10),FF(10,11),DTEST,ZTEST,PARAM
      WRITE (6,10) PARAM
   10 FORMAT (//,10X,'PARAM =',E14.7)
      DO 20 I=1,M
   20 WRITE (6,21) I,ZPERM(I)
   21 FORMAT (10X,I2,3X,E14.7)
      WRITE (6,22)
   22 FORMAT (//,1X)
      RETURN
      END
```

MAJOR PROGRAM -- ODD BUCKLING OF AN ELLIPSOIDAL SHELL.

```
C       MAIN ROUTINE
        COMMON DELTA,BEE,BEESQ,AITCH,RHO,GNU,ZTEST,DTEST,
      C  C1(40,40),C2(40,40),C3(40),S6(40,40),S7(40),
      C  S8(40,40,40),S9(40,40),E(2,40),FF(40,41),
      C  ZTEMP(40),ZPERM(40),P(40),B(40),RPERM,RESID
C    READ IN DATA.
        READ (5,27) XX,CREMEN
        IOTA=XX
        READ (5,27) X
     27 FORMAT (E14.7,66X)
        M=X
        READ (5,27) RHO,GNU,BEE,AITCH
        ZTEST=.75E-04
        DTEST=.6E-04
        DELTA=.0001
        JFLAG=0
        START=DELTA
        READ (5,28) (B(I),I=1,M)
     28 FORMAT (4(1X,E14.7))
      1 BEESQ=BEE**2
        CALL READIN(M)
C    MAIN LOOP STARTS HERE.
      2 DO 40 I=1,M
     40 ZPERM(I)=B(I)
        IF (IOTA.LT.1)       GO TO 41
        TEMPOR=RHO
        RHO=ZPERM(IOTA)
        ZPERM(IOTA)=TEMPOR
     41 CALL SECANT(M,IOTA)
        IF (IOTA.LT.1)       GO TO 42
        TEMPOR=RHO
        RHO=ZPERM(IOTA)
        ZPERM(IOTA)=TEMPOR
     42 IF (RPERM.GT.(.5E-02)) STOP
        CALL GRAPH(M,JFLAG,IOTA)
        JFLAG=1
        ZTEST=AMAX1(.5*RPERM,.75E-04)
        DELTA=START
        DO 73 I=1,M
     73 IF (ABS(B(I)).LT.(.1E-30)) B(I)=0
C     A PARAMETER IS VARIED AND THE PROGRAM IS CYCLED BACK.
        IF (IOTA.LT.1)       GO TO 238
    237 B(IOTA)=B(IOTA)+CREMEN
        GO TO 2
    238 RHO=RHO+CREMEN
        GO TO 2
        END
```

MAJOR PROGRAM -- ODD BUCKLING OF AN ELLIPSOIDAL SHELL.

```
      SUBROUTINE READIN(M)
      COMMON DELTA,BEE,BEESQ,AITCH,RHO,GNU,ZTEST,DTEST,
     C  C1(40,40),C2(40,40),C3(40),S6(40,40),S7(40),
     C  S8(40,40,40),S9(40,40),E(2,40),FF(40,41),
     C  ZTEMP(40),ZPERM(40),P(40),B(40),KPERM,RESID
C  THIS ROUTINE READS IN THE INTEGRALS.
C     THE LOOP LIMITS ARE SPECIAL TO THE CASE
C     M=40  IN THE PROGRAM "INTEGRALS" WHICH
C     PRODUCES THE VALUES BEING READ IN HERE.
      DO 200 I=1,40
  200 READ (8,201) I,C3(I),S7(I)
  201 FORMAT (I2,E14.7,E14.7,50X)
      DO 210 I=1,40
      DO 210 J=1,40
  210 READ (8,202) I,J,C1(I,J),C2(I,J),S6(I,J),S9(I,J)
  202 FORMAT (I2,I2,E14.7,E14.7,E14.7,E14.7,20X)
      DO  220  N=1,2870
      READ (8,203) I1,J1,K1,V1,I2,J2,K2,V2,I3,J3,K3,V3,
     C    I4,J4,K4,V4
  215 S8(I1,J1,K1)=V1
      S8(I1,K1,J1)=V1
      S8(J1,I1,K1)=V1
      S8(J1,K1,I1)=V1
      S8(K1,I1,J1)=V1
      S8(K1,J1,I1)=V1
  216 S8(I2,J2,K2)=V2
      S8(I2,K2,J2)=V2
      S8(J2,I2,K2)=V2
      S8(J2,K2,I2)=V2
      S8(K2,I2,J2)=V2
      S8(K2,J2,I2)=V2
  217 S8(I3,J3,K3)=V3
      S8(I3,K3,J3)=V3
      S8(J3,I3,K3)=V3
      S8(J3,K3,I3)=V3
      S8(K3,I3,J3)=V3
      S8(K3,J3,I3)=V3
  218 S8(I4,J4,K4)=V4
      S8(I4,K4,J4)=V4
      S8(J4,I4,K4)=V4
      S8(J4,K4,I4)=V4
      S8(K4,I4,J4)=V4
      S8(K4,J4,I4)=V4
  220 CONTINUE
  203 FORMAT(4(I2,I2,I2,E14.7))
      RETURN
      END
```

MAJOR PROGRAM -- ODD BUCKLING OF AN ELLIPSOIDAL SHELL.

```
      SUBROUTINE RESIDU(NNN,M,IOTA)
      COMMON DELTA,BEE,BEESQ,AITCH,RHO,GNU,ZTEST,DTEST,
     C C1(40,40),C2(40,40),C3(40),S6(40,40),S7(40),
     C S8(40,40,40),S9(40,40),E(2,40),FF(40,41),
     C ZTEMP(40),ZPERM(40),P(40),B(40),RPERM,RESID
C   THIS ROUTINE CALCULATES RPERM AND E(1,J) OR IT
C     CALCULATES RTEMP AND E(2,J) FOR THE BENEFIT OF SECANT.
C     THE VARIABLE KKK  TURNS OUT TO BE 1 IF I+J IS ODD,
C     2 IF I+J IS EVEN.  WHEN THE DEEPEST LOOP STARTS FROM
C     KKK AND GOES BY TWOS, MANY USELESS MULTIPLICATIONS BY
C     ZERO ARE ELIMINATED, AND THE SPEED OF THE SUBROUTINE
C     IS DOUBLED.
      IF (IOTA.LT.1) GO TO 41
      TEMPOR=RHO
      RHO=ZTEMP(IOTA)
      ZTEMP(IOTA)=TEMPOR
   41 CALL PCOMP(M,IOTA)
      FACTOR=12*(1-GNU**2)*BEESQ/(AITCH**3)
      DO 110 I=1,M
      EPHEM=0.0
      TRANS=0.0
      DO 108 J=1,M
      TEMP=0.0
      KKK=2-(I+J-((I+J)/2)*2)
      DO 105 K=KKK,M,2
  105 TEMP=TEMP+S8(I,J,K)*P(K)
      TRANS=TRANS+C1(I,J)*B(J)
  108 EPHEM=EPHEM-S6(I,J)*P(J)-(.5*RHO*S9(I,J)-TEMP)*B(J)
  110 E(NNN,I)=-TRANS+FACTOR*(-.5*RHO*S7(I)+EPHEM)
      RESID=0.0
      DO 120 I=1,M
  120 RESID=RESID+E(NNN,I)**2
      RESID=RESID**.5
      IF (IOTA.LT.1) RETURN
      TEMPOR=RHO
      RHO=ZTEMP(IOTA)
      ZTEMP(IOTA)=TEMPOR
      RETURN
      END
```

MAJOR PROGRAM -- ODD BUCKLING OF AN ELLIPSOIDAL SHELL.

```
      SUBROUTINE PCOMP(M,IOTA)
      COMMON DELTA,BEE,BEESQ,AITCH,RHO,GNU,ZTEST,DTEST,
     C  C1(40,40),C2(40,40),C3(40),S6(40,40),S7(40),
     C  S8(40,40,40),S9(40,40),E(2,40),FF(40,41),
     C  ZTEMP(40),ZPERM(40),P(40),B(40),RPERM,RESID
C   THIS ROUTINE COMPUTES THE P'S, GIVEN THE B'S, FOR THE
C       BENEFIT OF THE ROUTINES RESIDU AND GRAPH.  FOR
C       THE MEANING AND USE OF KKK, SEE THE COMMENT FOR
C       THE ROUTINE RESIDU.
      DO 5 I=1,M
    5 B(I)=ZTEMP(I)
      DO 10 I=1,M
      DO 10 J=1,M
   10 FF(I,J)=C2(I,J)
      DO 20 I=1,M
   20 FF(I,M+1)=RHO*C3(I)
      DO 30 I=1,M
      EPHEM=0.0
      DO 27 J=1,M
      TEMP=0.0
      KKK=2-(I+J-((I+J)/2)*2)
      DO 25 K=KKK,M,2
   25 TEMP=TEMP+S8(I,J,K)*B(K)
   27 EPHEM=EPHEM+(S6(I,J)-.5*TEMP)*B(J)
   30 FF(I,M+1)=FF(I,M+1)+AITCH*BEESQ*EPHEM
      CALL GAUSS(M,KS)
      DO 40 I=1,M
   40 P(I)=FF(I,M+1)
      RETURN
      END
```

```
      SUBROUTINE GRAPH(M,JFLAG,IOTA)
      COMMON DELTA,BEE,BEESQ,AITCH,RHO,GNU,ZTEST,UTEST,
     C  C1(40,40),C2(40,40),C3(40),S6(40,40),S7(40),
     C  S8(40,40,40),S9(40,40),E(2,40),FF(40,41),
     C  ZTEMP(40),ZPERM(40),P(40),B(40),RPERM,RESID
      DIMENSION XPOS(52),YPOS(52)
C   REPORT THE SOLUTION AND SOME OF THE PARAMETERS.
      WRITE (6,11) BEE,AITCH,GNU,IOTA
   11 FORMAT (//,1X,'BEE = ',E14.7,',    AITCH = ',E14.7,',
                                          GNU = ',
     C  E14.7,',    IOTA =',I3)
      WRITE (6,14) RPERM
   14 FORMAT (1X,'RESIDUAL =',E14.7)
      DO 15 I=1,M
      ZTEMP(I)=ZPERM(I)
   15 B(I)=ZPERM(I)
      CALL PCOMP(B,IOTA)
      WRITE (6,16) (B(I),I=1,M)
   16 FORMAT (4(1X,E14.7))
C   CALCULATE THE NEW POSITIONS OF FIFTY-THREE POINTS ON THE
C   ELLIPSOID.  IF JFLAG=0, WHICH IS SO THE FIRST TIME THIS
C   ROUTINE IS CALLED, REPORT THEM.  IN ANY CASE REPORT
C   RHO, POLAR DEFLECTION, AND EQUATORIAL CONTRACTION.
      IFLAG=0
      FACTOR=AITCH**3/(12.0*(1.0-GNU**2))
      RATIO=1/BEESQ-1
      F=1./AITCH
      L=2
      PI=3.141593
      DX=PI/(52*L)
      LL=52*L
 1000 WOA=0
      DO 100 J=1,LL
      XI=J*DX
      CO=COS(XI)
      SI=SIN(XI)
      DENOM=BEESQ*CO*CO+SI*SI
      SQRTD=SQRT(DENOM)
      X=SI/SQRTD
      Y=-BEESQ*CO/SQRTD
      ROALPH=SI+RATIO*SI**3
      R2=SI*SI/DENOM
      BETA=0.0
      PSI=0.0
      PSIP=0.0
      DO 10 I=1,M
      CS=COS(I*XI)
      SN=SIN(I*XI)
```

MAJOR PROGRAM -- ODD BUCKLING OF AN ELLIPSOIDAL SHELL.

```
      BETA=BETA+B(I)*SN
      PSI=PSI+P(I)*SN
      PSIP=PSIP+I*P(I)*CS
   10 CONTINUE
      PHI=XI-BETA
      SP=SIN(PHI)
      CP=COS(PHI)
      RZEROV=-.5*RHO*SI*SI/DENOM
      TEMP=SP/(ROALPH*AITCH)*(X*AITCH+PSI*CP+RZEROV*SP-ROALP
                                             H*GNU*PSIP-
     C   R2*GNU*RHO*SP)-BEESQ*SI/(SQRTD**3)
      TEMP=TEMP*DX
      WOA=WOA+TEMP
      IF (J.GT.(J/L)*L) GO TO 100
      XPOS(J/L)=X+F*(ROALPH*PSIP-R2*RHO*SP-GNU*PSI*CP-GNU*RZ
                                             EROV*SP)
      YPOS(J/L)=Y+WOA-TEMP/2
  100 CONTINUE
      IF (IFLAG.EQ.1) GO TO 108
      DEFLEC = BFE - YPOS(52)
      CONTRA=1.-XPOS(26)
      WRITE (6,104) RHO,DEFLEC,CONTRA
  104 FORMAT (1X,'RHO =',E14.7,'  DEFLECTION =',F8.5,'  CONT
                                             RACTION ='
     C   ,F8.5)
      IF (JFLAG.NE.0) RETURN
      WRITE (6,107)  YPOS(52)
  107 FORMAT (/,1X,'DEFORMED SHAPE:              NORTH POLE AT   '
                                             ,F8.4)
  108 IF  (IFLAG.EQ.1)  WRITE (6,109)
  109 FORMAT (1X,'DETAIL NEAR SOUTH POLE')
      DO 110 I=1,17
  110 WRITE (6,111) XPOS(I),YPOS(I),XPOS(17+I),YPOS(17+I),XP
                                             OS(34+I),
     C      YPOS(34+I)
  111 FORMAT (1X,3(F8.4,2X,F8.4,4X))
      WRITE (6,200)
  200 FORMAT (//,20X)
      DX=PI/(2704*L)
      IFLAG=IFLAG+1
      IF (IFLAG.LT.2) GO TO 1000
      RETURN
      END
```

MAJOR PROGRAM -- EVEN BUCKLING OF AN ELLIPSOIDAL SHELL.

```
C     MAIN ROUTINE
      COMMON DELTA,BEE,BEESQ,AITCH,RHO,GNU,ZTEST,DTEST,
     C C1(20,20),C2(20,20),C3(20),S6(20,20),S7(20),
     C S8(20,20,20),S9(20,20),E(2,20),FF(20,21),
     C ZTEMP(20),ZPERM(20),P(20),B(20),RPERM,RESID
C  READ IN DATA.
      READ (5,27) XX,CREMEN
      IOTA=XX
      READ (5,27) X
   27 FORMAT (E14.7,66X)
      M=X
      READ (5,27) RHO,GNU,BEE,AITCH
      ZTEST=.75E-04
      DTEST=.6E-04
      DELTA=.0001
      JFLAG=0
      START=DELTA
      READ (5,28) (B(I),I=1,M)
   28 FORMAT (4(1X,F14.7))
    1 BEESQ=BEE**2
      CALL READIN(M)
C  MAIN LOOP STARTS HERE.
    2 DO 40 I=1,M
   40 ZPERM(I)=B(I)
      IF (IOTA.LT.1)      GO TO 41
      TEMPOR=RHO
      RHO=ZPERM(IOTA)
      ZPERM(IOTA)=TEMPOR
   41 CALL SECANT(M,IOTA)
      IF (IOTA.LT.1)      GO TO 42
      TEMPOR=RHO
      RHO=ZPERM(IOTA)
      ZPERM(IOTA)=TEMPOR
   42 IF (RPERM.GT.(.5E-02)) STOP
      CALL GRAPH(M,JFLAG,IOTA)
      JFLAG=1
      ZTEST=AMAX1(.5*RPERM,.75E-04)
      DELTA=START
      DO 73 I=1,M
   73 IF (ABS(B(I)).LT.(.1E-30)) B(I)=0
C   A PARAMETER IS VARIED AND THE PROGRAM IS CYCLED BACK.
      IF (IOTA.LT.1)      GO TO 238
  237 B(IOTA)=B(IOTA)+CREMEN
      GO TO 2
  238 RHO=RHO+CREMEN
      GO TO 2
      END
```

MAJOR PROGRAM -- EVEN BUCKLING OF AN ELLIPSOIDAL SHELL.

```
      SUBROUTINE READIN(M)
      COMMON DELTA,BEE,BEESQ,AITCH,RHO,GNU,ZTEST,OTEST,
     C  C1(20,20),C2(20,20),C3(20),S6(20,20),S7(20),
     C  S8(20,20,20),S9(20,20),E(2,20),FF(20,21),
     C  ZTEMP(20),ZPERM(20),P(20),B(20),KPERM,RESID
C   THIS ROUTINE READS IN THE INTEGRALS ,  OMITTING
C        ALL THOSE WHICH HAVE ANY ODD SUBSCRIPTS.
C        THE LOOP LIMITS ARE SPECIAL TO THE CASE M=40
C        IN THE PROGRAM "INTEGRALS" WHICH PRODUCES THE
C        VALUES BEING READ IN HERE.
      DO 200 I=1,40
      READ (8,201)  J,X,XX
      IF ((J/2)*2.LT.J) GO TO 200
      J=J/2
      C3(J)=X
      S7(J)=XX
  200 CONTINUE
  201 FORMAT (I2,E14.7,E14.7,50X)
      DO 210 I=1,40
      DO 210 J=1,40
      READ (8,202)    I1,JJ,X,XX,XXX,XXXX
      IF ((II/2)*2.LT.II.OR.(JJ/2)*2.LT.JJ) GO TO 210
      II=II/2
      JJ=JJ/2
      C1(II,JJ)=X
      C2(II,JJ)=XX
      S6(II,JJ)=XXX
      S9(II,JJ)=XXXX
  210 CONTINUE
  202 FORMAT (I2,I2,E14.7,E14.7,E14.7,E14.7,20X)
      DO  220  N=1,2870
      READ (8,203) I1,J1,K1,V1,I2,J2,K2,V2,I3,J3,K3,V3,
     C    I4,J4,K4,V4
      IF ((I1/2)*2.LT.I1.OR.(J1/2)*2.LT.J1.OR.(K1/2)*2.LT.K1
                                                ) GO TO 216
      I1=I1/2
      J1=J1/2
      K1=K1/2
      S8(I1,J1,K1)=V1
      S8(I1,K1,J1)=V1
      S8(J1,I1,K1)=V1
      S8(J1,K1,I1)=V1
      S8(K1,I1,J1)=V1
      S8(K1,J1,I1)=V1
  216 IF ((I2/2)*2.LT.I2.OR.(J2/2)*2.LT.J2.OR.(K2/2)*2.LT.K2
                                                ) GO TO 217
      I2=I2/2
      J2=J2/2
```

```
      K2=K2/2
      S8(I2,J2,K2)=V2
      S8(I2,K2,J2)=V2
      S8(J2,I2,K2)=V2
      S8(J2,K2,I2)=V2
      S8(K2,I2,J2)=V2
      S8(K2,J2,I2)=V2
  217 IF (((I3/2)*2.LT.I3.OR.(J3/2)*2.LT.J3.OR.(K3/2)*2.LT.K3
     1                                                 ) GO TO 218
      I3=I3/2
      J3=J3/2
      K3=K3/2
      S8(I3,J3,K3)=V3
      S8(I3,K3,J3)=V3
      S8(J3,I3,K3)=V3
      S8(J3,K3,I3)=V3
      S8(K3,I3,J3)=V3
      S8(K3,J3,I3)=V3
  218 IF (((I4/2)*2.LT.I4.OR.(J4/2)*2.LT.J4.OR.(K4/2)*2.LT.K4
     1                                                 ) GO TO 220
      I4=I4/2
      J4=J4/2
      K4=K4/2
      S8(I4,J4,K4)=V4
      S8(I4,K4,J4)=V4
      S8(J4,I4,K4)=V4
      S8(J4,K4,I4)=V4
      S8(K4,I4,J4)=V4
      S8(K4,J4,I4)=V4
  220 CONTINUE
  203 FORMAT(4(I2,I2,T2,E14.7))
      RETURN
      END
```

```
      SUBROUTINE RESIDU(NNN,M,IOTA)
      COMMON DELTA,BEE,BEESQ,AITCH,RHO,GNU,ZTEST,DTEST,
     C C1(20,20),C2(20,20),C3(20),S6(20,20),S7(20),
     C S8(20,20,20),S9(20,20),E(2,20),FF(20,21),
     C ZTEMP(20),ZPERM(20),P(20),B(20),RPERM,RESID
C  THIS ROUTINE CALCULATES RPERM AND E(1,J) OR IT
C     CALCULATES RTEMP AND E(2,J) FOR THE BENEFIT OF SECANT.
      IF (IOTA.LT.1) GO TO 41
      TEMPOR=RHO
      RHO=ZTEMP(IOTA)
      ZTEMP(IOTA)=TEMPOR
   41 CALL PCOMP(M,IOTA)
      FACTOR=12*(1-GNU**2)*BEESQ/(AITCH**3)
      DO 110 I=1,M
      EPHEM=0.0
      TRANS=0.0
      DO 108 J=1,M
      TEMP=0.0
      DO 105 K=1,M
  105 TEMP=TEMP+S8(I,J,K)*P(K)
      TRANS=TRANS+C1(I,J)*B(J)
  108 EPHEM=EPHEM-S6(I,J)*P(J)-(.5*RHO*S9(I,J)-TEMP)*B(J)
  110 E(NNN,I)=-TRANS+FACTOR*(-.5*RHO*S7(I)+EPHEM)
      RESID=0.0
      DO 120 I=1,M
  120 RESID=RESID+E(NNN,I)**2
      RESID=RESID**.5
      IF (IOTA.LT.1) RETURN
      TEMPOR=RHO
      RHO=ZTEMP(IOTA)
      ZTEMP(IOTA)=TEMPOR
      RETURN
      END
```

MAJOR PROGRAM -- EVEN BUCKLING OF AN ELLIPSOIDAL SHELL.

```
      SUBROUTINE PCOMP(M,IOTA)
      COMMON DELTA,BEE,BEESQ,AITCH,RHO,GNU,ZTEST,DTEST,
     C  C1(20,20),C2(20,20),C3(20),S6(20,20),S7(20),
     C  S8(20,20,20),S9(20,20),E(2,20),FF(20,21),
     C  ZTEMP(20),ZPERM(20),P(20),B(20),RPERM,RESID
C   THIS ROUTINE COMPUTES THE P'S, GIVEN THE B'S, FOR THE
C      BENEFIT OF THE ROUTINES RESIDU AND GRAPH.
      DO 5 I=1,M
    5 B(I)=ZTEMP(I)
      DO 10 I=1,M
      DO 10 J=1,M
   10 FF(I,J)=C2(I,J)
      DO 20 I=1,M
   20 FF(I,M+1)=RHO*C3(I)
      DO 30 I=1,M
      EPHEM=0.0
      DO 27 J=1,M
      TEMP=0.0
      DO 25 K=1,M
   25 TEMP=TEMP+S8(I,J,K)*B(K)
   27 EPHEM=EPHEM+(S6(I,J)-.5*TEMP)*B(J)
   30 FF(I,M+1)=FF(I,M+1)+AITCH*BEESQ*EPHEM
      CALL GAUSS(M,KS)
      DO 40 I=1,M
   40 P(I)=FF(I,M+1)
      RETURN
      END
```

```
      SUBROUTINE GRAPH(M,JFLAG,IOTA)
      COMMON DELTA,BEE,BEESQ,AITCH,RHO,GNU,ZTEST,DTEST,
     C C1(20,20),C2(20,20),C3(20),S6(20,20),S7(20),
     C S8(20,20,20),S9(20,20),F(2,20),FF(20,21),
     C ZTEMP(20),ZPERM(20),P(20),B(20),RPERM,RESID
      DIMENSION XPOS(52),YPOS(52)
C  REPORT THE SOLUTION AND SOME OF THE PARAMETERS.
      WRITE (6,11) BEE,AITCH,GNU,IOTA
   11 FORMAT (//,1X,'BEE = ',E14.7,',    AITCH = ',E14.7,',
                                              GNU = ',
     C E14.7,',    IOTA =',I3)
      WRITE (6,14) RPERM
   14 FORMAT (1X,'RESIDUAL =',E14.7)
      DO 15 I=1,M
      ZTEMP(I)=ZPERM(I)
   15 B(I)=ZPERM(I)
      CALL PCOMP(M,IOTA)
      WRITE (6,16) (B(I),I=1,M)
   16 FORMAT (4(1X,E14.7))
C  CALCULATE THE NEW POSITIONS OF FIFTY-THREE POINTS ON THE
C  ELLIPSOID.  IF JFLAG=0, WHICH IS SO THE FIRST TIME THIS
C  ROUTINE IS CALLED, REPORT THEM.  IN ANY CASE REPORT
C  RHO, POLAR DEFLECTION, AND EQUATORIAL CONTRACTION.
      IFLAG=0
      FACTOR=AITCH**3/(12.0*(1.0-GNU**2))
      RATIO=1/BEESQ-1
      F=1./AITCH
      L=2
      PI=3.141593
      DX=PI/(52*L)
      LL=52*L
 1000 WOA=0
      DO 100 J=1,LL
      XI=J*DX
      CO=COS(XI)
      SI=SIN(XI)
      DENOM=BEESQ*CO*CO+SI*SI
      SQRTD=SQRT(DENOM)
      X=SI/SQRTD
      Y=-BEESQ*CO/SQRTD
      ROALPH=SI+RATIO*SI**3
      R2=SI*SI/DENOM
      BETA=0.0
      PSI=0.0
      PSIP=0.0
      DO 10 I=1,M
      CS=COS(I*XI)
      SN=SIN(I*XI)
```

MAJOR PROGRAM -- EVEN BUCKLING OF AN ELLIPSOIDAL SHELL.

```
      BETA=BETA+B(I)*SN
      PSI=PSI+P(I)*SN
      PSIP=PSIP+2*I*P(I)*CS
   10 CONTINUE
      PHI=XI-BETA
      SP=SIN(PHI)
      CP=COS(PHI)
      RZEROV=-.5*RHO*SI*SI/DENOM
      TEMP=SP/(ROALPH*AITCH)*(X*AITCH+PSI*CP+RZEROV*SP-ROALP
                                                  H*GNU*PSIP-
     C   R2*GNU*RHO*SP)-BEESQ*SI/(SQRTD**3)
      TEMP=TEMP*DX
      WOA=WOA+TEMP
      IF (J.GT.(J/L)*L) GO TO 100
      XPOS(J/L)=X+F*(ROALPH*PSIP-R2*RHO*SP-GNU*PSI*CP-GNU*RZ
                                                  EROV*SP)
      YPOS(J/L)=Y+WOA-TEMP/2
  100 CONTINUE
      IF (IFLAG.EQ.1) GO TO 108
      DEFLEC = BEE - YPOS(52)
      CONTRA=1.-XPOS(26)
      WRITE (6,104) RHO,DEFLEC,CONTRA
  104 FORMAT (1X,'RHO =',E14.7,'  DEFLECTION =',F8.5,'  CONT
                                                  RACTION ='
     C   ,F8.5)
      IF (JFLAG.NE.0) RETURN
      WRITE (6,107)  YPOS(52)
  107 FORMAT (/,1X,'DEFORMED SHAPE:          NORTH POLE AT  '
                                                  ,F8.4)
  108 IF   (IFLAG.EQ.1)  WRITE (6,109)
  109 FORMAT (1X,'DETAIL NEAR SOUTH POLE')
      DO 110 I=1,17
  110 WRITE (6,111) XPOS(I),YPOS(I),XPOS(17+I),YPOS(17+I),XP
                                                  OS(34+I),
     C   YPOS(34+I)
  111 FORMAT (1X,3(F8.4,2X,F8.4,4X))
      WRITE (6,200)
  200 FORMAT (//,20X)
      DX=PI/(2704*L)
      IFLAG=IFLAG+1
      IF (IFLAG.LT.2) GO TO 1000
      RETURN
      END
```

MAJOR PROGRAM -- ODD BUCKLING OF A SPHERICAL SHELL.

```
C      MAIN ROUTINE
       COMMON RHOS,AOH,GNU,ROOT,DELTA,ZTEST,RPERM,RESID,
      C  RCOE,ELAOH,DTEST,TABLE(35,35,35),E(2,35),FF(35,36),
      C  ZTEMP(35),ZPERM(35),A(35),B(35)
C   READ IN DATA
       READ (5,27) X,CREMEN
       IOTA=X
       READ (5,27) X
       M=X
       READ (5,27) RHOS,GNU,AOH
       DELTA=.0001
       KOUNT=0
       START=DELTA
       DTEST=.6E-04
       ZTEST=.25E-04
       READ (5,28) (A(I),I=1,M)
       ROOT=(3.-3.*GNU**2)**.5
       RCOE = (2. / (AOH*AOH*ROOT)) * (1.+GNU/(2.*AOH*ROOT))
       ELAOH=1.+GNU/(2.*AOH*ROOT)
    27 FORMAT (E17.10,63X)
    28 FORMAT (4(1X,F14.7,5X))
C   READ IN TABLE
       DO 10 I=1,35
       DO 10 J=1,35
       DO 10 K =1,35
    10 TABLE(I,J,K)=0.0
       REWIND 8
       DO 20 L=1,10850
       READ (8,21) I,J,K,VALUE
       IF (I.GT.35.OR.J.GT.35.OR.K.GT.35) GO TO 20
       TABLE(I,J,K)=VALUE
       TABLE(I,K,J)=VALUE
       TABLE(J,I,K)=VALUE
       TABLE(J,K,I)=VALUE
       TABLE(K,I,J)=VALUE
       TABLE(K,J,I)=VALUE
    20 CONTINUE
       END FILE 8
    21 FORMAT (I2,I2,I2,E14.7)
C   THE MAIN LOOP STARTS HERE. SET UP ZPERM AND CALL SECANT
     1 DO 40 I=1,M
    40 ZPERM(I)=A(I)
       IF (IOTA.LT.1) GO TO 50
       TEMP=RHOS
       RHOS=ZPERM(IOTA)
       ZPERM(IOTA)=TEMP
    50 CALL SECANT(M,IOTA)
       IF (IOTA.LT.1) GO TO 60
```

MAJOR PROGRAM -- ODD BUCKLING OF A SPHERICAL SHELL.

```
      TEMP=RHOS
      RHOS=ZPERM(IOTA)
      ZPERM(IOTA)=TEMP
   60 CALL OUTPUT (M,IOTA)
C  RE-SET SECANT'S PARAMETERS
      KOUNT=0
      IF (RPERM.GT.(.25E-03)) STOP
      ZTEST=AMAX1(.5*RPERM,.25E-04)
      DELTA=START
C    A PARAMETER IS VARIED AND THE PROGRAM IS CYCLED BACK.
      IF (IOTA.LT.1) GO TO 200
      A(IOTA)=A(IOTA)+CREMEN
      GO TO 1
  200 RHOS=RHOS+.003
      GO TO 1
      END
```

MAJOR PROGRAM -- ODD BUCKLING OF A SPHERICAL SHELL.

```
      SUBROUTINE COMPB  (M,IOTA)
      COMMON RHOS,AOH,GNU,ROOT,DELTA,ZTEST,RPERM,RESID,
     C RCOF,ELAOH,DTEST,TABLE(35,35,35),E(2,35),FF(35,36),
     C ZTEMP(35),ZPERM(35),A(35),B(35)
C    COMPUTE THE B'S.  THE LOOP LIMITS ARE SET UP
C    TO AVOID UNNECESSARY MULTIPLICATIONS BY ZERO.
      RATIO=AOH*ROOT/(2*ELAOH)
      DO 2 I=1,M
    2 A(I)=ZTEMP(I)
      DO 200 N=1,M
      TN1=2*N+1
      TNN1=2*N*(N+1)
      T=0
      DO 100 L=1,M
      IA=IABS(L-N)
      IB=1.5+.5*(-1)**IA
      IC=MAX0(IA,IB)
      ID=MIN0(M,L+N)
      DO 100 K=IC,ID,2
  100 T = T + TABLE(L,K,N)*TN1*A(L)*A(K)/TNN1
  200 B(N)=RATIO/(N*(N+1)-1-GNU)*(T/2-A(N))
      RETURN
      END
```

MAJOR PROGRAM -- ODD BUCKLING OF A SPHERICAL SHELL.

```
      SUBROUTINE RESIDU  (NNN,M,IOTA)
      COMMON RHOS,AOH,GNU,ROOT,DELTA,ZTEST,RPERM,RESID,
     C RCOE,ELAOH,DTEST,TABLE(35,35,35),E(2,35),FF(35,36),
     C ZTEMP(35),ZPERM(35),A(35),B(35)
C   COMPUTE THE RESIDUAL AND THE VECTOR E FOR
C   THE USE OF THE SUBROUTINE SECANT.  THE LOOP
C   LIMITS ARE SET UP SO THAT UNNECESSARY
C   MULTIPLICATIONS BY ZERO MAY BE AVOIDED.
      IF (IOTA.LT.1) GO TO 1
      TEMP=RHOS
      RHOS=ZTEMP(IOTA)
      ZTEMP(IOTA)=TEMP
    1 CALL COMPB(M,IOTA)
      FOUR=4*ROOT*ELAOH*AOH
      DO 200 N=1,M
      TN1=2*N+1
      TNN1=2*N*(N+1)
      T=0
      DO 100 L=1,M
      IA=IABS(L-N)
      IB=1.5+.5*(-1)**IA
      IC=MAX0(IA,IB)
      ID=MIN0(M,L+N)
      DO 100 K=IC,ID,2
  100 T = T + TABLE(L,K,N)*TN1*A(L)*B(K)/TNN1
  200 E(NNN,N)=((N*(N+1)-1+GNU)/FOUR-RHOS)
     C        *A(N)-2*B(N)+2*T
      RESID=0.0
      DO 300 I=1,M
  300 RESID=RESID+E(NNN,I)**2
      RESID=RESID**.5
      IF (IOTA.LT.1) RETURN
      TEMP=RHOS
      RHOS=ZTEMP(IOTA)
      ZTEMP(IOTA)=TEMP
      RETURN
      END
```

MAJOR PROGRAM -- ODD BUCKLING OF A SPHERICAL SHELL.

```
      SUBROUTINE OUTPUT (M,IOTA)
      COMMON RHOS,AOH,GNU,ROOT,DELTA,ZTEST,RPERM,RESID,
     C  RCOF,FLAOH,DTEST,TABLE(35,35,35),E(2,35),FF(35,36),
     C  ZTEMP(35),ZPERM(35),A(35),B(35)
      DO 10 I=1,M
      A(I)=ZPERM(I)
   10 ZTEMP(I)=ZPERM(I)
      CALL COMPB(M,IOTA)
      WRITE (6,11) RHOS,AOH,GNU
   11 FORMAT (1X,'RHOS =',E14.7,'     AOH =',F8.3,'    GNU =',F
                                        6.3)
      WRITE (6,21) (A(I),I=1,M)
   21 FORMAT (4(1X,F14.7,5X))
      CALL DEFLEC (M)
      RETURN
      END
```

MAJOR PROGRAM -- ODD BUCKLING OF A SPHERICAL SHELL.

```
      SUBROUTINE DEFLEC (M)
      COMMON RHOS,AOH,GNU,ROOT,DELTA,ZTEST,RPERM,RESID,
     C RCOE,ELAOH,DTEST,TABLE(35,35,35),E(2,35),FF(35,36),
     C ZTEMP(35),ZPERM(35),A(35),B(35)
C     COMPUTE POLAR DEFLECTION.
      W=2-RHOS*RCOE*AOH*(.8*GNU*A(2)+(1-GNU))
      DO 10 L=2,M,2
   10 W=W-2*(A(L)-RCOE*AOH*B(L)*(1+GNU)-(1-GNU)*RHOS*RCOE*AO
                                                  H*A(L)/2)
      DO 20 N=1,M
   20 W=W-N*(N+1.)*A(N)**2/(2.*N+1.)
      T=0
      DO 30 K=1,M
      KMINUS=(K-1)-((K-1)/2)*2
      DO 30 N=1,M
      NMINUS=(N-1)-((N-1)/2)*2
      NTOP=(N-1+NMINUS)/2
      KTOP=(K-1+KMINUS)/2
      U=-K*N*PRINN(K,N)
      DO 28 J=1,NTOP
   28 U=U-K*(4*J-2*NMINUS-1)*PRINN(K,2*J-2*NMINUS-1)
      DO 27 J=1,KTOP
   27 U=U-N*(4*J-2*KMINUS-1)*PRINN(N,2*J-2*KMINUS-1)
      DO 26 I=1,KTOP
      DO 26 J=1,NTOP
   26 U=U-(4*I-2*KMINUS-1)*(4*J-2*NMINUS-1)*
     C PRINN(2*I-2*KMINUS-1,2*J-2*NMINUS-1)
   30 T=T+U*B(K)*A(N)
      W=W+(1+GNU)*AOH*RCOE*T
      T=0
      DO 40 K=1,M
      DO 40 N=1,M
      NMINUS=(N-1)-((N-1)/2)*2
      NTOP=(N-1+NMINUS)/2
      U=-PRINN(K,N)
      DO 38 I=1,NTOP
   38 U=U-(4*I-2*NMINUS-1)*PRINN(K,2*I-NMINUS-1)
   40 T=T+U*B(K)*A(N)*K*(K+1)*N
      W=W-GNU*AOH*RCOE*T
      T=0
      DO 50 K=1,M
      DO 50 N=1,M
      U=-PRINN(K-1,N+1)+PRINN(K-1,N-1)+PRINN(K+1,N+1)-PRINN(
                                                   K+1,N-1)
      NPARTY=N-(N/2)*2
      NTOP=(N+NPARTY)/2
      DO 48 I=1,NTOP
   48 U=U+K*(K+1)*(4*I-2*NPARTY-1)*(-PRINN(K-1,2*I-NPARTY-1)
```

MAJOR PROGRAM -- ODD BUCKLING OF A SPHERICAL SHELL.

```
  C    *PRINN(K+1,2*I-NPARTY-1))/(2*K+1)
50 T=T-U*A(K)*A(N)*K*N*(K+1)*(N+1)/((2*K+1.0)*(2*N+1.0))
   W=W+RHOS*AOH*RCOE*T
   T=0
   DO 60 N=1,M
60 T=T+2.*N*(N+1.)*A(N)*B(N)/(2.*N+1.)
   W=W+RCOE*AOH*T
   W2=2-W
   W3=W2/2
   WRITE (6,70) W,W2,W3
70 FORMAT (1X,'DISTANCE BETWEEN POLES = ',F8.5,
  C        /,1X,'TOTAL POLAR DEFLECTION = ',F8.5,
  C        /,1X,'SEMI POLAR DEFLECTION  = ',F8.5)
   WRITE (6,80)
80 FORMAT (16(58X,'****',/))
   RETURN
   END
```

```
      FUNCTION PRINN(K,N)
C     INNER PRODUCT OF LEGENDRE FUNCTIONS K AND N.
      PRINN=0.0
      IF (N.NE.K) RETURN
      PRINN=2./(2.*N+1.)
      RETURN
      END
```

MAJOR PROGRAM -- EVEN BUCKLING OF A SPHERICAL SHELL.

```
C     MAIN ROUTINE
      COMMON RHOS,AOH,GNU,ROOT,DELTA,ZTEST,RPERM,RESID,
     C E(2,30),FF(30,31),ZTEMP(30),ZPERM(30),A(30),B(30),
     C TABLE(30,30,30),DTEST,RCOE,ELAOH
C   READ IN DATA
      READ (5,27) X,CREMEN
      IOTA=X
      READ (5,27) X
      M=X
      READ (5,27) RHOS,GNU,AOH
      DELTA=.0001
      DTEST=.60E-04
      START=DELTA
      ZTEST=.25E-04
      READ (5,28) (A(I),I=1,M)
      ROOT=(3.-3.*GNU**2)**.5
      RCOE = (2. / (AOH*AOH*ROOT)) * (1.+GNU/(2.*AOH*ROOT))
      ELAOH=1.+GNU/(2.*AOH*ROOT)
   27 FORMAT (E17.10,63X)
   28 FORMAT (4(1X,E14.7,5X))
C   READ IN TABLE
   11 DO 10 I=1,M
      DO 10 J=1,M
      DO 10 K =1,M
   10 TABLE(I,J,K)=0.0
      REWIND 8
      DO 20 L=1,10850
      READ (8,21) I,J,K,VALUE
      IF (I.GT.2*M.OR.J.GT.2*M.OR.K.GT.2*M) GO TO 20
      IF ((I/2)*2.LT.I.OR.(J/2)*2.LT.J.OR.(K/2)*2.LT.K) GO T
                                                       O 20
      TABLE(I/2,J/2,K/2)=VALUE
      TABLE(I/2,K/2,J/2)=VALUE
      TABLE(J/2,I/2,K/2)=VALUE
      TABLE(J/2,K/2,I/2)=VALUE
      TABLE(K/2,I/2,J/2)=VALUE
      TABLE(K/2,J/2,I/2)=VALLE
   20 CONTINUE
      END FILE 8
   21 FORMAT (I2,I2,I2,E14.7)
C   THE MAIN LOOP STARTS HERE. SET UP ZPERM AND CALL SECANT.
    1 DO 40 I=1,M
   40 ZPERM(I)=A(I)
      IF (IOTA.LT.1) GO TO 47
      TEMP=RHOS
      RHOS=ZPERM(IOTA)
      ZPERM(IOTA)=TEMP
   47 CALL SECANT(M,IOTA).
```

MAJOR PROGRAM -- EVEN BUCKLING OF A SPHERICAL SHELL.

```
      IF (IOTA.LT.1) GO TO 49
      TEMP=RHOS
      RHOS=ZPERM(IOTA)
      ZPERM(IOTA)=TEMP
   49 CALL OUTPUT (M,IOTA)
C     RE-SET SECANT'S PARAMETERS
      IF (RPERM.GT.(.25E-03)) STOP
      ZTEST=AMAX1(.5*RPERM,.25E-04)
      DELTA=START
C     A PARAMETER IS VARIED AND THE PROGRAM IS CYCLED BACK.
      IF (IOTA.LT.1) GO TO 51
      A(IOTA)=A(IOTA)+CREMEN
      GO TO 1
   51 RHOS=RHOS+CREMEN
      GO TO 1
      END
```

MAJOR PROGRAM -- EVEN BUCKLING OF A SPHERICAL SHELL.

```
      SUBROUTINE COMPB  (M,IOTA)
      COMMON RHOS,AOH,GNU,ROOT,DELTA,ZTEST,RPERM,RESID,
     C E(2,30),FF(30,31),ZTEMP(30),ZPERM(30),A(30),B(30),
     C TABLE(30,30,30),DTEST,RCOE,ELAOH
C     COMPUTE THE B'S.  THE LOOP LIMITS ARE SET
C     UP TO AVOID UNNECESSARY MULTIPLICATIONS BY ZERO.
      RATIO=AOH*ROOT/(2*ELAOH)
      DO 2 I=1,M
    2 A(I)=ZTEMP(I)
      DO 200 N=1,M
      NN=N+N
      TN1=2*NN+1
      TNN1=2*NN*(NN+1)
      T=0
      DO 100 L=1,M
      IA=IABS(L-N)
      IC=MAX0(IA,1)
      ID=MIN0(M,L+N)
      DO 100 K=IC,ID
  100 T = T + TABLE(L,K,N)*TN1*A(L)*A(K)/TNN1
  200 B(N)=RATIO/(NN*(NN+1)-1-GNU)*(T/2-A(N))
      RETURN
      END
```

MAJOR PROGRAM -- EVEN BUCKLING OF A SPHERICAL SHELL.

```
      SUBROUTINE RESIDU  (NNN,M,IOTA)
      COMMON RHOS,AOH,GNU,ROOT,DELTA,ZTEST,RPERM,RESID,
     C E(2,30),FF(30,31),ZTEMP(30),ZPERM(30),A(30),B(30),
     C TABLE(30,30,30),DTEST,RCOE,ELAOH
      IF (IOTA.LT.1) GO TO 50
C    COMPUTE THE RESIDUAL AND THE VECTOR E FOR
C    THE USE OF THE PROGRAM SECANT.  THE LOOP
C    LIMITS ARE SET UP TO AVOID UNNECESSARY
C    MULTIPLICATIONS BY ZERO.
      TEMP=RHOS
      RHOS=ZTEMP(IOTA)
      ZTEMP(IOTA)=TEMP
   50 CALL COMPB(M,IOTA)
      FOUR=4*ROOT*ELAOH*AOH
      DO 200 N=1,M
      NN=N+N
      TN1=2*NN+1
      TNN1=2*NN*(NN+1)
      T=0
      DO 100 L=1,M
      IA=IABS(L-N)
      IC=MAX0(IA,1)
      ID=MIN0(M,L+N)
      DO 100 K=IC,ID
  100 T = T + TABLE(L,K,N)*TN1*A(L)*B(K)/TNN1
  200 E(NNN,N)=((NN*(NN+1)-1+GNU)/FOUR-RHOS)
     C         *A(N)-2*B(N)+2*T
      RESID=0.0
      DO 300 I=1,M
  300 RESID=RESID+E(NNN,I)**2
      RESID=RESID**.5
      IF (IOTA.LT.1) RETURN
      TEMP=RHOS
      RHOS=ZTEMP(IOTA)
      ZTEMP(IOTA)=TEMP
      RETURN
      END
```

MAJOR PROGRAM -- EVEN BUCKLING OF A SPHERICAL SHELL.

```
      SUBROUTINE OUTPUT (M,IOTA)
      COMMON RHOS,AOH,GNU,ROOT,DELTA,ZTEST,RPERM,RESID,
     C E(2,30),FF(30,31),ZTEMP(30),ZPERM(30),A(30),B(30),
     C TABLE(30,30,30),DTEST,RCOE,ELAOH
      DO 10 I=1,M
      A(I)=ZPERM(I)
   10 ZTEMP(I)=ZPERM(I)
      CALL COMPR(M,IOTA)
      WRITE (6,11) RHOS,AOH,GNU
   11 FORMAT (1X,'RHOS =',E14.7,'    AOH =',F8.3,'    GNU =',F
                                                 6.3)
      WRITE (6,21) (A(I),I=1,M)
   21 FORMAT (4(1X,E14.7,5X))
      CALL DEFLEC (M)
      RETURN
      END
```

MAJOR PROGRAM -- EVEN BUCKLING OF A SPHERICAL SHELL.

```
      SUBROUTINE DEFLEC (M)
      COMMON RHOS,AOH,GNU,ROOT,DELTA,ZTEST,RPERM,RESID,
     C E(2,30),FF(30,31),ZTEMP(30),ZPERM(30),A(30),B(30),
     C TABLE(30,30,30),DTEST,RCOE,ELAOH
      INTEGER S,STEP
C     COMPUTE POLAR DEFLECTION.
      S=M
      STEP=2
      W=2-RHOS*RCOE*AOH*(.8*GNU*A(2/2)+(1-GNU))
      DO 10 L=2,S,2
   10 W=W-2*(A(L/2)-RCOE*AOH*B(L/2)*(1+GNU)-(1-GNU)*RHOS*RCO
                                                   E*AOH
     C           *A(L/2)/2)
      DO 20 N=STEP,S,STEP
   20 W=W-N*(N+1.)*A(N/2)**2/(2.*N+1.)
      T=0
      DO 30 K=STEP,S,STEP
      KMINUS=(K-1)-((K-1)/2)*2
      DO 30 N=STEP,S,STEP
      NMINUS=(N-1)-((N-1)/2)*2
      NTOP=(N-1+NMINUS)/2
      KTOP=(K-1+KMINUS)/2
      U=-K*N*PRINN(K,N)
      DO 28 J=1,NTOP
   28 U=U-K*(4*J-2*NMINUS-1)*PRINN(K,2*J-2*NMINUS-1)
      DO 27 J=1,KTOP
   27 U=U-N*(4*J-2*KMINUS-1)*PRINN(N,2*J-2*KMINUS-1)
      DO 26 I=1,KTOP
      DO 26 J=1,NTOP
   26 U=U-(4*I-2*KMINUS-1)*(4*J-2*NMINUS-1)*
     C    PRINN(2*I-2*KMINUS-1,2*J-2*NMINUS-1)
   30 T=T+U*B(K/2)*A(N/2)
      W=W+(1+GNU)*AOH*RCOE*T
      T=0
      DO 40 K=STEP,S,STEP
      DO 40 N=STEP,S,STEP
      NMINUS=(N-1)-((N-1)/2)*2
      NTOP=(N-1+NMINUS)/2
      U=-PRINN(K,N)
      DO 38 I=1,NTOP
   38 U=U-(4*I-2*NMINUS-1)*PRINN(K,2*I-NMINUS-1)
   40 T=T+U*B(K/2)*A(N/2)*K*(K+1)*N
      W=W-GNU*AOH*RCOE*T
      T=0
      DO 50 K=STEP,S,STEP
      DO 50 N=STEP,S,STEP
      U=-PRINN(K-1,N+1)+PRINN(K-1,N-1)+PRINN(K+1,N+1)-PRINN(
                                                   K+1,N-1)
```

MAJOR PROGRAM -- EVEN BUCKLING OF A SPHERICAL SHELL.

```
      NPARTY=N-(N/2)*2
      NTOP=(N+NPARTY)/2
      DO 48 I=1,NTOP
48    U=U+K*(K+1)*(4*I-2*NPARTY-1)*(-PRINN(K-1,2*I-NPARTY-1)
     C     +PRINN(K+1,2*I-NPARTY-1))/(2*K+1)
50    T=T-U*A(K/2)*A(N/2)*K*N*(K+1)*(N+1)/((2*K+1.0)*(2*N+1.
                                                          0))
      W=W+RHOS*AOH*RCOE*T
      T=0
      DO 60 N=STEP,S,STEP
60    T=T+2.*N*(N+1.)*A(N/2)*B(N/2)/(2.*N+1.)
      W=W+RCOE*AOH*T
      W2=2-W
      W3=W2/2
      WRITE (6,70) W,W2,W3
70    FORMAT (1X,'DISTANCE BETWEEN POLES = ',F8.5,
     C        /,1X,'TOTAL POLAR DEFLECTION = ',F8.5,
     C        /,1X,'SEMI POLAR DEFLECTION  = ',F8.5)
      WRITE (6,80)
80    FORMAT (16(38X,'***',/))
      RETURN
      END
```

MAJOR PROGRAM -- EVEN BUCKLING OF A SPHERICAL SHELL.

```
      FUNCTION PRINN(K,N)
C     COMPUTE INNER PRODUCT OF LEGENDRE POLYNOMIALS K AND N.
      PRINN=0.0
      IF (N.NE.K) RETURN
      PRINN=2./(2.*N+1.)
      RETURN
      END
```

SATELLITE PROGRAM -- INTEGRALS FOR ELLIPSOID PROBLEM.

```
C       MAIN ROUTINE
        DIMENSION  C1(40,40),C2(40,40),C3(40),S6(40,40),
       C  S7(40),S8(40,40,40),S9(40,40),T1(321),T2(321),
       C  T3(321),T4(321),S5(40,40),SINE(40)
        BEE=.5
        BEESQ=BEE*BEE
        GNU=.3
        M=40
        MM=3*M+1
        MMM=200+MM
C   STEP 1A: COMPUTE INTEGRALS OF SIN (I*XI).
        DO 1 I=1,MMM
      1 T1(I)=0.0
        DO 2 II=1,MM,2
        I=200+II
        J=200-II
        T1(I)=2.0/II
      2 T1(J)=-T1(I)
C   STEP 1B: COMPUTE INTEGRALS OF KERNEL * SIN (I*XI)
C   FOR A WIDE RANGE OF VALUES OF I, USING SIMPSON'S RULE.
C   THE VARIABLE JJJ BELOW IS 4 WHEN J IS ODD,
C   2 WHEN J IS EVEN.
        PI=3.141593
        DX=PI/1000
        DO 5 I=1,MMM
        T2(I)=0
        T3(I)=0
      5 T4(I)=0
        DO 6 I=1,MM,2
        J=(-1)**((I+3)/2)
        T2(200+I)=J
        T3(200+I)=J
      6 T4(200+I)=J
        DO 10 J=1,499
        JJJ=2+2*(J-2*(J/2))
        XI=J*DX
        SN=SIN(XI)
        CS=COS(XI)
        DENOM=BEESQ*CS**2+SN**2
        DENOM2=DENOM**2
        DENOM3=DENOM**1.5
        DENOM4=DENOM3*DENOM
        DO 10 II=201,MMM,2
        I=II-200
        SI=SIN(I*XI)
        T2(II)=T2(II)+JJJ*SI/DENOM3
        T3(II)=T3(II)+JJJ*SI/DENOM4
     10 T4(II)=T4(II)+JJJ*SI/DENOM2
```

```
      DO 12 II=201,MMM,2
      T2(II)=T2(II)*DX*.6666667
      T3(II)=T3(II)*DX*.6666667
      T4(II)=T4(II)*DX*.6666667
      I=II-200
   12 CONTINUE
      DO 18 II=1,MM
      I=200-II
      J=200+II
      T2(I)=-T2(J)
      T3(I)=-T3(J)
   18 T4(I)=-T4(J)
C     STEP 2A: COMPUTE THE INTEGRALS S5(I,J) DIRECTLY,
C     USING SIMPSON'S RULE.    THE VARIABLE KKK BELOW
C     IS 4 WHEN K IS ODD, 2 WHEN K IS EVEN.
      DO 40 I=1,M
      DO 40 J=1,M
   40 S5(I,J)=0.0
      DO 80 K=1,499
      XI=K*DX
      KKK=2+2*(K-(K/2)*2)
      SN=SIN(XI)
      CS=COS(XI)
      CSSQ=CS*CS
      DENOM=BEESQ*CSSQ+SN*SN
      DO 60 J=1,M
   60 SINE(J)=SIN(J*XI)
      RATIO=0.0
      DO 80 I=1,M
      RATIO=RATIO*CS+COS((I-1)*XI)
      DO 80 J=I,M,2
   80 S5(I,J)=S5(I,J)+KKK*CSSQ*RATIO*SINE(J)/DENOM
      DO 90 I=1,M
      DO 90 J=I,M,2
      S5(J,I)=S5(I,J)*DX*.6666667
      S5(I,J)=S5(J,I)
   90 CONTINUE
   91 FORMAT (3X,2(I2,3X),E14.7)
C     STEP 2B: COMPUTE THE INTEGRALS S1-S4, S6-S12,
C     AND THEIR COMBINATIONS C1,C2, AND C3, AS DESCRIBED
C     IN THE SECTION ON EVALUATION OF INTEGRALS.
      DO 110  II=1,M
      DO 110 J=1,M
      I=II+200
      S1=-(T1(I+J+1)-T1(I+J-1)-T1(I-J+1)+T1(I-J-1))/4
      S2=(T1(I+J+3)-3*T1(I+J+1)+3*T1(I+J-1)-T1(I+J-3)
     C   -T1(I-J+3)+3*T1(I-J+1)-3*T1(I-J-1)+T1(I-J-3))/8
      S3=(T1(I+J+1)+T1(I+J-1)+T1(I-J+1)+T1(I-J-1))/4
```

SATELLITE PROGRAM -- INTEGRALS FOR ELLIPSOID PROBLEM.

```
      S4=-(T1(I+2+J+1)+T1(I+2+J-1)+T1(I+2-J+1)+T1(I+2-J-1)
     C     -2*(T1(I+J+1)+T1(I+J-1)+T1(I-J+1)+T1(I-J-1))
     C     +T1(I-2+J+1)+T1(I-2+J-1)+T1(I-2-J+1)+T1(I-2-J-1))/
                                                       16
      C1(II,J)=-J**2*(S1+(1-BEESQ)/BEESQ*S2)
     C             +J*(S3+3*(1-BEESQ)/BEESQ*S4)
     C              -(BEESQ*S5(II,J) + GNU*S1)
  110 C2(II,J)=C1(II,J) +  2*GNU*S1
      DO 120 II=1,M
      I=II+200
      S10=-(T4(I+2+1)+T4(I+2-1)-2*(T4(I+1)+T4(I-1))
     C          +T4(I-2+1)+T4(I-2-1))/8
      S11=-(T4(I+2+3)+3*(T4(I+2+1)+T4(I+2-1))+T4(I+2-3)
     C     -2*(T4(I+3)+3*(T4(I+1)+T4(I-1))+T4(I-3))
     C        +T4(I-2+3)+3*(T4(I-2+1)+T4(I-2-1))+T4(I-2-3))/32
      S12=(T4(I+4+1)+T4(I+4-1)-4*(T4(I+2+1)+T4(I+2-1))
     C     +6*(T4(I+1)+T4(I-1))-4*(T4(I-2+1)+T4(I-2-1))
     C     +T4(I-4+1)+T4(I-4-1))/32
  120 C3(II)=-.5*BEESQ*S10+BEESQ*(3-.5*GNU)*S11+(1-.5*GNU+2*
                                                       BEESQ)*S12
      DO 130 II=1,M
      I=II+200
      DO 130 J=1,M
      DO 125 K=1,M
  125 S8(II,J,K)=-(T2(I+J+K+1)+T2(I+J+K-1)-T2(I+J-K+1)-T2(I+
                                                       J-K-1)
     C          -T2(I-J+K+1)-T2(I-J+K-1)+T2(I-J-K+1)+T2(I-J-K-1
                                                       ))/8
      S6(II,J)=-(T2(I+J+1)-T2(I+J-1)-T2(I-J+1)+T2(I-J-1))/4
      S9(II,J)=(T3(I+J+3)-3*T3(I+J+1)+3*T3(I+J-1)-T3(I+J-3)
     C            -T3(I-J+3)+3*T3(I-J+1)-3*T3(I-J-1)+T3(I-J-3))
                                                       /16
  130 S7(II)=-(T3(I+2+1)+T3(I+2-1)-2*(T3(I+1)+T3(I-1))
     C          +T3(I-2+1)+T3(I-2-1))/8
      DO 200 I=1,M
  200 WRITE (6,201) I,C3(I),S7(I)
  201 FORMAT (I2,E14.7,E14.7,50X)
      DO 210 I=1,M
      DO 210 J=1,M
  210 WRITE (6,202) I,J,C1(I,J),C2(I,J),S6(I,J),S9(I,J)
  202 FORMAT (I2,I2,E14.7,E14.7,E14.7,E14.7,20X)
      WRITE (6,203) ((((I,J,K,S8(I,J,K)),I=1,J),J=1,K),K=1,M
                                                       )
  203 FORMAT(4(I2,I2,I2,E14.7))
      STOP
      END
```

SATELLITE PROGRAM -- INTEGRALS FOR SPHERE PROBLEM.

```
C     MAIN ROUTINE
      INTEGER HIEST,PARL,PARN,STEP,SS,SSP,SSM
      DOUBLE PRECISION U,V,A(200)
      LOWEST=1
      HIEST=51
      STEP=1
      MA=2*HIEST
      MAP=MA+1
      SS=4*HIEST
      SSP=SS+1
      SSM=SS-1
      A(MA)=1
      DO 60 I=MAP,SS
      J=I-MA
      A(J)=0
   60 A(I)=(2*I-SSP)*A(I-1)/(I-MA)
      A(MA)=1
      KAPPA=0
      DO 200 L=LOWEST,HIEST,STEP
      DO 200 M=STEP,L,STEP
      DO 200 N=STEP,M,STEP
      PARL=L-(L/2)*2
      LA=(L+PARL)/2
      IF ((L+M+N).GT.((L+M+N)/2)*2) GO TO 200
      IF (L.GT.(M+N)) GO TO 200
      PARN=N-(N/2)*2
      NA=(N-PARN)/2
      V=0
      DO 150 I=1,LA
      IA=4*I-1-2*PARL
      IB=2*I-1-PARL
      IC=M-1
      ID=(IB+IC+N)/2+MA
      IE=M+1
      IF=(IB+IE+N)/2+MA
      U=0
      IF (NA.EQ.0) GO TO 150
      DO 140 J=1,NA
      JA=4*J-3+2*PARN
      JB=2*J-2+PARN
      JC=(IB+IC+JB)/2+MA
      JD=(IB+IE+JB)/2+MA
  140 U=U+JA*2*(A(JC-IB)*A(JC-IC)*A(JC-JB)/((2*JC-SSM)*A(JC)
                                                      )-
     C     A(JD-IB)*A(JD-IE)*A(JD-JB)/((2*JD-SSM)*A(JD)))
  150 V=V+IA*(N*2*(A(ID-IB)*A(ID-IC)*A(ID-N)/((2*ID-SSM)*A(I
                                                      D))-
     C      A(IF-IB)*A(IF-IE)*A(IF-N)/((2*IF-SSM)*A(IF)))+U)
```

SATELLITE PROGRAM -- INTEGRALS FOR SPHERE PROBLEM.

```
    V=M*(M+1)*V/(2*M+1)
    KAPPA=KAPPA+1
    WRITE (8,199) L,M,N,V
199 FORMAT (I2,I2,I2,F14.7)
200 CONTINUE
    WRITE (6,201) KAPPA
201 FORMAT (20X,I6,' RECORDS WRITTEN')
    STOP
    END
```

SATELLITE PROGRAM -- MERIDIAN FOR SPHERE PROBLEM.

```
C      MAIN ROUTINE
       DIMENSION XPOS(100),YPOS(100),A(50),BS(50)
       INTEGER S,STEP
       REAL NU
       DP(N,X)=N*(P(N-1,X)-X*P(N,X))/((1-X*X)**.5)
       DDP(N,X)=N*((-1-N+N*X*X)*P(N,X)+X*P(N-1,X))/(X*X-1)
       READ (5,10) AOH,NU,RHOS,X,XX
       S=X
       STEP=XX
       READ (5,10) (A(I),I=STEP,S,STEP)
       READ (5,10) (BS(I),I=STEP,S,STEP)
    10 FORMAT (E17.10,63X)
       ROOT=(3-3*NU**2)**.5
       RCOE=(2./(AOH**2*ROOT))*(1+NU/(2*AOH*ROOT))
       F=RCOE*AOH
       PI=3.141593
       L=4
       DX=PI/(100*L)
       LL=100*L-1
       WOA=0
       DO 100 I=1,LL
       XI=I*DX
       Y=COS(XI)
       BETA=0
       PSIS=0
       PSISP=0
       DO 90 N=STEP,S,STEP
       TEMP=DP(N,Y)
       BETA=BETA+A(N)*TEMP
       PSIS=PSIS+BS(N)*TEMP
    90 PSISP=PSISP+BS(N)*DDP(N,Y)
       X=SIN(XI)
       Y=-Y
       SXI=X
       CXI=-Y
       TEMP=-BETA*CXI-(1-NU)/2*RHOS*F*SXI+F*PSIS*CXI-NU*F*PSI
                                                         SP*SXI
     C   +(1-NU)/2*RHOS*F*BETA*CXI-NU*RHOS*F*BETA*SXI**2*CXI
     C   -.5*BETA**2*SXI-F*PSIS*BETA*CXI**2/SXI+NU*F*PSISP*BE
                                                         TA*CXI
     C   +NU*RHOS*F*BETA**2*SXI*CXI**2+F*PSIS*BETA*SXI
       TEMP=DX*TEMP
       WOA=WOA+TEMP
       IF (I.GT.(I/L)*L) GO TO 100
       XPOS(I/L)=X+F*((NU-1)/2*RHOS*SXI+PSISP*SXI-NU*PSIS*CXI
                                                      +RHOS*BETA
     C   *SXI**2*CXI)
       YPOS(I/L)=Y+WOA-TEMP/2
```

AD-A032 752 CITY UNIV OF NEW YORK GRADUATE SCHOOL AND UNIV CENTER F/G 13/13
DEEP BUCKLING OF A THIN OBLATE SPHEROIDAL SHELL UNDER UNIFORM N--ETC(U)
1976 N H JACOBS AF-AFOSR-2063-71

UNCLASSIFIED AFOSR-TR-76-1209 NL

2 OF 2
ADA032752

END
DATE
FILMED
1 -77

SATELLITE PROGRAM -- MERIDIAN FOR SPHERE PROBLEM.

```
100 CONTINUE
    YPOS(100)=1+WOA
    ADJUST=(1-YPOS(100))/2
    DO 105 I=1,100
105 YPOS(I)=YPOS(I)+ADJUST
    DEFLEC=2*(1-YPOS(100))
    DIST=2*YPOS(100)
    WRITE (6,1000) DIST,DEFLEC
1000 FORMAT (1X,'DIST =',E14.7,/,1X,'DEFLEC =',E14.7)
    STOP
    END
```

SATELLITE PROGRAM -- MERIDIAN FOR SPHERE PROBLEM.

```
      FUNCTION P(N,X)
C   THE N-TH LEGENDRE POLYNOMIAL EVALUATED AT X.
      P=1
      IF (N.EQ.1) P=X
      IF (N.LT.2) RETURN
      S=1
      T=X
      DO 10 I=2,N
      R=S
       S=T
   10 T=((2*I-1)*X*S-(I-1)*R)/I
      P=T
      RETURN
      END
```

SATELLITE PROGRAM -- FROM SPHERE TO ELLIPSOID.

```
C        MAIN ROUTINE
C     A SOLUTION FOR BETA, EXPRESSED IN TERMS OF
C        LEGENDRE FUNCTIONS, IS TRANSLATED INTO
C        A SOLUTION FOR BETA, EXPRESSED IN TERMS OF SINES.
         INTEGER SS,STEP,SNEW
         REAL NU,LAOH
         DIMENSION Z(20),A(20)
         DP(N,X)=N*(P(N-1,X)-X*P(N,X))/((1-X*X)**.5)
         DO 1 I=1,20
       1 Z(I)=0.0
         READ (5,2) RHOST,NU,AOH,SS,STEP,SNEW
       2 FORMAT (3E15.7,3I2)
         READ (5,3) (A(I),I=STEP,SS,STEP)
       3 FORMAT (E15.7)
         SQT=(3*(1-NU**2))**.5
         LAOH=1+NU/((2*AOH)*SQT)
         RHOCRT=2*LAOH/(SQT*AOH*AOH)
         RHO=RHOST*RHOCRT
         X=SNEW
         BEE=1
         AITCH=BEE/AOH
         WRITE (9,28) X,RHO,NU,BEE,AITCH
         DX=3.141593/100
      10 DO 20 M=1,99
         X=M*DX
         XI=COS(X)
         DO 20 N=STEP,SS,STEP
         Y=DP(N,XI)
         DO 20 I=1,SNEW
         SINE=SIN(I*X)
      20 Z(I)=Z(I)+A(N)*Y*SINE
         DO 30 I=1,SNEW
         A(I)=Z(I)*2*DX/3.141593
         WRITE (9,28) A(I)
      28 FORMAT (E14.7)
      30 CONTINUE
         STOP
         END
```

SATELLITE PROGRAM -- FROM SPHERE TO ELLIPSOID.

```
      FUNCTION P(N,X)
C   THE N-TH LEGENDRE POLYNOMIAL EVALUATED AT X.
      P=1
      IF (N.EQ.1) P=X
      IF (N.LT.2) RETURN
      S=1
      P=X
      DO 10 I=2,N
      R=S
      S=P
   10 P=((2*I-1)*X*S-(I-1)*R)/I
      RETURN
      END
```

Bibliography

[1] R. A. Clark and E. Reissner, "On Stresses and Deformations of Ellipsoidal Shells Subject to Internal Pressure", Journal of the Mechanics and Physics of Solids, 1957, Vol. 6, pp. 63-70.

[2] D. A. Danielson, "Buckling and Initial Post-buckling Behavior of Spheroidal Shells under Pressure", AIAA Journal, 1969, Vol. 7, pp. 936-944.

[3] IBM, System/360 Scientific Subroutine Package, Version III: Programmer's Manual, 5th ed., 1970.

[4] E. Polak, "A Globally Converging Secant Method with Applications to Boundary Value Problems", SIAM Journal of Numerical Analysis, 1974, Vol. 11, pp. 529-537.

[5] H. E. Rauch, "Instability of Thin-Walled Spherical Structures under External Pressure", in Contributions to Analysis: A Collection of Papers Dedicated to Lipman Bers, 1974, pp. 357-373.

[6] H. E. Rauch, N. H. Jacobs, and J. L. Mars, "Buckling of Complete Spherical Shells under Uniform External Pressure", to be submitted.

[7] Eric Reissner, "On the Theory of Thin Elastic

Shells", in <u>H. Reissner Anniversary Volume: Contributions to Applied Mechanics</u>, 1949, pp. 231-247.

[8] Eric Reissner, "On Axisymmetrical Deformations of Thin Shells of Revolution", in <u>Symposium in Applied Mathematics: Proceedings</u>, 1950, Vol. 3, pp. 27-52.